GERENCIAMENTO DE TEXTURAS PARA APLICAÇÕES DE VISUALIZAÇÃO DE TERRENOS EM AMBIENTE DE COMANDO E CONTROLE

Renato Massayuki Okamoto

MONOGRAFIA SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA O EXAME DE QUALIFICAÇÃO AO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Aprovada por:

____

Prof. Cláudio Esperança, Ph. D.
Orientador

____

Prof. Antonio Alberto Fernandes de Oliveira, D. Sc.

____

Prof. Sérgio Luiz Cardoso Salomão, D. Sc.

RIO DE JANEIRO, RJ - BRASIL

JUNHO DE 2004

Monografia submetida à COPPE como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.).

GERENCIAMENTO DE TEXTURAS PARA APLICAÇÕES DE VISUALIZAÇÃO DE TERRENOS EM AMBIENTE DE COMANDO E CONTROLE

Renato Massayuki Okamoto

Junho - 2004

Orientador: Prof. Cláudio Esperança, Ph. D.

Programa de Engenharia em Sistemas e Computação
Laboratório de Computação Gráfica

Key Words: Gerenciamento de Texturas, Visualização de Terrenos, Comando e Controle, Visão Computacional, Computação Gráfica.

Resumo:

Este trabalho aborda o gerenciamento de textura como aspecto crucial para a manipulação de conjuntos de grandes imagens raster, mais especificamente mapas cartográficos, associados a uma malha gerada a partir de um DEM, para fins de visualização de terreno em um ambiente informatizado para auxílio a tomada de decisões (comando e controle).

ÍNDICE

CAPÍTULO 1

INTRODUÇÃO

1.1– MOTIVAÇÃO

Muitas aplicações, que vão desde Sistemas de Informações Geográficas em 3D até simuladores de vôo, utilizam-se de modelos tridimensionais de terreno. Um sistema que forneça suporte a esses tipos de aplicações de forma interativa deve possuir um mecanismo de renderização em tempo real de grandes conjuntos de dados de terreno.

A evolução da tecnologia da informação se faz cada vez mais presente nas modernas forças armadas. As operações militares, que antes eram planejadas e controladas em papel, passam a ser controladas por meios eletrônicos através de sistemas de comando e controle [Camp92].

A necessidade de conhecer o terreno sempre foi uma habilidade essencial ao comandante militar. O conhecimento do terreno através do mapa em papel vem sendo utilizado por estrategistas militares a muitos séculos. Nos dias atuais a demanda pelo próximo salto tecnológico é a digitalização do campo de batalha.

O aumento do poder computacional e a disseminação de poderosos hardwares de renderização motivaram a criação de aplicações de ambientes virtuais visualmente ricos e bem realísticos. Naturalmente houve o interesse de empregar essas tecnologias para a visualização do campo de batalha.

Embora exista grande quantidade de trabalhos que buscam simplificar superfícies poligonais, existia pouca pesquisa no desenvolvimento de técnicas mais eficazes na utilização de texturas. Porém, com o aumento da capacidade gráfica dos hardwares atuais e a constante evolução das Unidades de Processamento Gráficas (GPU – *Graphic Processing Units*), este cenário está mudando, havendo um gradual aumento de interesse na área de processamento de texturas.

1.2– OBJETIVOS

A proposta deste trabalho é apresentar os fundamentos para desenvolver uma ferramenta para o gerenciamento de texturas baseadas em mapas de terrenos a ser utilizando em um ambiente interativo de visualização do campo de batalha.

Sendo assim, o estudo de abordagens atuais de manipulação de texturas será utilizado para propor um modelo de gerenciamento de múltiplas texturas em multiresolução em um ambiente 2,5 D para fins de visualização de terreno.

1.3– ORGANIZAÇÃO DO TRABALHO

O capítulo 2 tem por objetivo conceituar Comando e Controle (C2)e realizar uma breve introdução sobre sistemas de C2 com ênfase na importância da visualização de terrenos. Ao final serão apresentados alguns sistemas de C2 em desenvolvimento ou em uso a título de exemplo.

No capítulo 3 serão apresentados alguns conceitos importantes quando se trata de manipulação de grandes geometrias associadas à modelagem de terrenos assim como algumas metodologias utilizadas para sua visualização. Serão resumidamente descritas algumas das metodologias de simplificação de malhas e algoritmos de níveis de detalhes mais conhecidos na área de visualização de terrenos.

O capítulo 4 vai discorrer de forma reduzida sobre o que é o mapeamento de texturas além de apresentar um sistema típico de carregamento de texturas. Abordará também a bem conhecida técnica do MIP-Map como ferramenta filtragem contra o *aliasing*.

Serão apresentadas algumas metodologias para o mapeamento e gerenciamento de grandes texturas no capítulo 5, com ênfase na manipulação de texturas para a visualização de terrenos.

Por fim, o último capítulo apresentará a proposta do trabalho a ser desenvolvido bem como uma breve conclusão.

CAPÍTULO 2

COMANDO E CONTROLE

Comando e Controle (C2) é o processo pelo qual os comandantes militares e gerentes civis exercitam a sua autoridade e direção sobre seus recursos materiais e humanos para alcançar objetivos estratégicos e táticos (Ince *et al.*, 1997).

Desta forma, as Organizações Militares (OMs) possuem um sistema de informações que atuam em várias áreas, dentre as quais podemos destacar: manobra, logística, apoio de fogo, inteligência e interoperabilidade com demais forças. Estas informações são armazenadas em Sistemas Gerenciadores de Banco de Dados, apoiadas por uma infra-estrutura de comunicações, que, em tese, deve ser adequada à característica de alta mobilidade dos componentes do sistema (Boyes e Andriole, 1987).

Em tais ambientes, os dados podem estar localizados ou na rede fixa ou nas estações móveis. Existem diferentes tipos de estações móveis. Alguns computadores podem ser extremamente simples, com capacidade limitada. Neste caso, os dados são retirados de computadores de maior capacidade, onde as estações recuperam os dados através de procedimentos de *downloading* de acordo com as suas necessidades. Mais interessante, entretanto, é o ambiente onde as estações são mais potentes e armazenam dados nativos e, possivelmente, compartilhados por outros usuários – tecnicamente designados "*walkstations*" (Imielinski e Korth, 1996).

Os *links* de comunicação entre os participantes de um ambiente de comando e controle são geralmente de baixa velocidade. Existe a necessidade das OMs consultarem ou atualizarem a base de dados das outras OMs as quais estejam relacionadas sobre, por exemplo, atualização das posições geográficas de unidades amigas e inimigas, dados de pessoal, material e atividades que foram, estejam ou serão executadas.

2.1 - CONCEITOS E DEFINIÇÕES

C2 é alcançado através da implementação orquestrada de uma série de facilidades ligadas a área de comunicações, controle de pessoal e de equipamento e procedimentos para a monitoração, previsão, planejamento, direção, alocação de recursos e geração de opções para alcançar objetivos gerais e específicos. Em organizações empresariais este processo pode ser denominado *gerenciamento operacional de negócios* (Ince *et al.*, 1997).

Comando, Controle, Comunicações e Informações (C^3I) é o sistema integrado, adaptado a doutrina, procedimentos, estrutura organizacional, pessoal, equipamentos e facilidades de comunicações que permitem às autoridades de todos os níveis elaborar

planos, dirigir e controlar as suas atividades e gerir os recursos disponíveis (Ince *et al.*, 1997).

Sistemas civis e militares de C^3I são similares em seus requisitos. O sucesso desses sistemas depende da sua possibilidade de produzir e executar decisões oportunamente com informação acurada e precisa. Na indústria, gerentes e líderes de corporação identificam objetivos de mercado e mobilizam os seus recursos para alcançá-los. No campo militar, chefes militares planejam e executam longas e complicadas operações para alcançarem seus objetivos. Diretores da indústria mobilizam linhas de produção, gerentes, trabalhadores especializados e seus recursos naturais e manufaturados para produzir produtos superiores. Comandantes militares mobilizam armas, tropas e sofisticados meios de comunicação para defender e conquistar territórios e objetivos militares e políticos.

O levantamento de requisitos do usuário é a chave para um projeto bem sucedido de qualquer Sistema de Informação, assim como para sistemas de C^3I. Neste sentido, o levantamento de requisitos é normalmente mais fácil na área militar do que em organizações civis devido à descrição e detalhes da maior parte das atividades militares estarem bem reguladas e definidas em termos de níveis de comando, conflito, missões chaves e categorias.

Poder-se-ia listar de maneira genérica os elementos básicos de qualquer sistema de C^3I, sejam eles em nível estratégico, tático, de teatro de operações, de forças policiais, e outros similares como (Oliveira, 1999):

- *Subsistemas de missões*, que reúnem informações sobre a localização, movimentos, atividades inimigas e posicionamento dos meios amigos;

- *Subsistemas de navegação*, que informam, às forças amigas, o posicionamento de seus meios;

- *Centros de Comando e Integração*, que reúnem, integram e mostram as atividades dos recursos amigos e inimigos, proporcionando a dinamização dos meios adequados para emprego imediato.

Em seu papel integrador, os Centros de Comando e Integração devem estar em condições de responder à complexidade e às variações rápidas das situações táticas, as quais impõem freqüentes limitações aos comandantes, para compreenderem a situação geral e particular que lhes apresentam, dando-lhes subsídios seguros para adotar uma decisão ótima e resposta apropriada.

O modo natural de implementar um sistema de C^3I é normalmente através da automatização dos procedimentos previstos em manuais e regulamentos, possibilitando a obtenção de resultados de uma maneira, hipoteticamente mais rápida e eficiente.

Um sistema de C^3I tático é iniciado com sensores que examinam o ambiente e geram as primeiras informações as quais são imediatamente avaliadas e comparadas aos

recursos disponíveis, permitindo desenvolver um **P**lano de **Op**erações (POp) (Oliveira, 1999).

O POp permite visualizar as atribuições e ações necessárias que serão distribuídas, de modo a obter o máximo rendimento dos recursos disponíveis. Essas ações afetam o ambiente, gerando novos dados para os sensores. A partir daí o circuito é fechado e cada elemento é realimentado. Para que o círculo tenha o seu funcionamento normal, é de se supor que seus componentes devam ser unidos por meios de enlace de comunicações tecnicamente confiáveis. Isto nem sempre ocorre, visto que as comunicações poderiam ser rompidas por problemas técnicos ou por interferência de ações do inimigo.

A Figura 2.1 permite verificar vários pontos interessantes. A similaridade com um sistema de controle automático é aparente, com exceção das funcionalidades ligadas à inteligência do sistema. Esta não é imediatamente receptiva a uma determinada quantificação, porém sua importância é tal que se assemelha muito aos sistemas de comando e controle enlaçados pelas comunicações. Pode-se citar como exemplo o sistema de C^3I, onde a letra I significa ***I**nteligência* e, nos níveis menores de comando, ***I**nformações*.
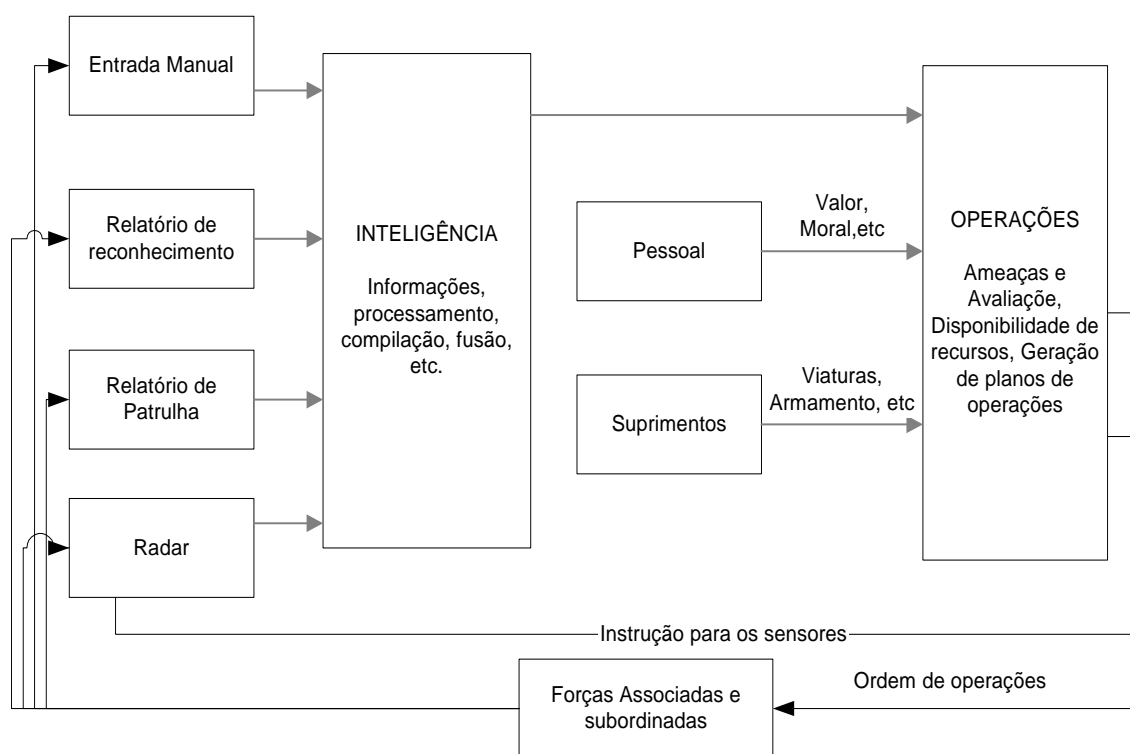
Figura 2.1 - Modelo de Comando e Controle Tático

Este melhor emprego das informações para a elaboração das decisões de comando aponta a possibilidade da utilização do sistema de C^3I como multiplicador de força ou do poder de combate. Este conceito pode ser modelado por meio da equação de *Lanchester*, que estabelece que o efetivo de uma força é proporcional ao produto da eficácia de suas armas pelo quadrado do seu quantitativo.

Numa situação hipotética, um exército em face de uma força numericamente superior, de dois para um, é necessário contê-la com uma arma que seja quatro vezes mais eficaz do que as forças inimigas, de modo a alcançar a igualdade. Exemplos do passado são as fortificações, onde apesar de possuir efetivo consideravelmente menor do que as forças agressoras, elas conseguiam resistir ao ataque em virtude da eficácia de sua construção.

Hodiernamente, a eficácia dos sistemas de C^3I é operacionalizado por meio da concentração ágil de forças nos locais de engajamento, de modo a se obter superioridade numérica localizada e, pela assimetria, no engajamento de armas.

O caminho mais eficaz para neutralizar um sistema de C^3I é cortar os enlaces ou nodos do sistema. Quando um único enlace de um sistema de C^3I é suprimido, é necessário que enlaces redundantes, já existentes, sejam imediatamente instalados para assegurar a sobrevivência do sistema. Não somente estes fatores, mas outros também devem permitir que se faça a atualização do sistema, quando da restauração das comunicações.

Os sistemas de C^3I táticos são considerados mais complexos e dinâmicos que os estratégicos. Exigem operações mais próximas ao terreno sob o controle inimigo e, por isso, devem ser totalmente móveis ou, pelo menos, transportáveis.

As operações táticas são caracterizadas pelas rápidas mudanças de posição e, em conseqüência, impõem severas limitações aos sistemas C^3I táticos. Estas limitações afetam a habilidade dos comandantes para perceber a situação geral, interagindo às atividades amigas e inimigas, avaliando as ameaças para, depois, decidir e comandar as respostas apropriadas.

2.2 - VISUALIZAÇÃO DO CAMPO DE BATALHA

Apesar do advento dos computadores e o desenvolvimento de softwares de apoio à decisão, os militares ainda utilizam mapas em papel e folhas de acetatos, chamados calcos, sobrepostas (Figura 2.2) para a visualização do campo de batalha e no planejamento das operações. Este é um legado dos dias em que os relatórios provenientes do campo chegavam aos centros de operações de combate exclusivamente via rádio. O rádio-operador recebia um relatório verbal que era então traduzido para uma simbologia manualmente desenhada em um mapa de papel.
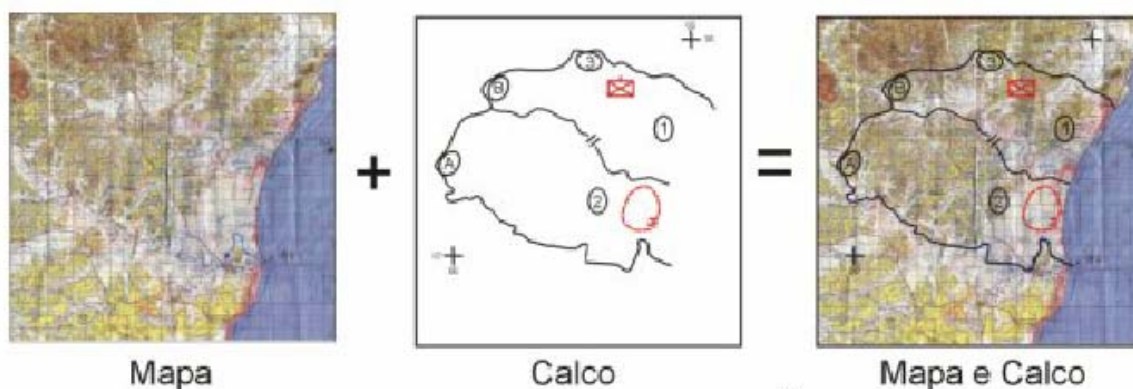
Figura 2.2 – Mapa e calco utilizados no planejamento das operações.

O processo de atualizar mapas e desenhar o planejamento de vários cenários é desgastante e consome grande quantidade de tempo. As modernas operações militares são altamente dinâmicas, produzindo informações críticas muito rapidamente de forma que as técnicas manuais tornam-se inadequadas para a correta visualização do campo de batalha.

Com o avanço da tecnologia, links digitais criptografados permitem que unidades de combate espacialmente distribuídas enviem relatórios e recebam ordens e informações utilizando redes de computadores ao invés da tradicional rede de rádios. Os dados estão disponíveis diretamente na forma digital.

O campo de batalha é uma arena grande e complexa. Um sistema de visualização para Comando e Controle está interessado em mostrar a situação de um teatro de operações e pode envolver a representação de áreas geográficas muito grandes e de milhares de unidades militares. Os comandantes militares estão interessados no posicionamento e organização de suas tropas, das forças inimigas e de elementos não combatentes. Se a operação for conjunta com outra forças - terra, ar e mar - ou mesmo de outras nacionalidades, o sistema de visualização deve mostrar se as diferentes forças estão operando de modo a dar suporte umas às outras (Feibush, Gagvani e Williams, 2000).

2.3 – EXEMPLOS DE SISTEMAS DE COMANDO E CONTROLE

Dada a natureza sigilosa do tema, poucos são os artigos e teses de domínio público descrevendo técnicas computacionais que combinam a visualização e a simulação na área militar.

Porém, pode-se perceber a importância do tema quando tomamos como exemplo os esforços dos EUA. A maior parte das pesquisas é mantida pelo governo americano, o qual prevê um gasto de cerca de US$ 4 bilhões por ano em simulação computacional para o treinamento de suas Forças Armadas (Macedonia, 2002). Nenhum outro país investe tanto nesta área.

No *US Naval Research Laboratory* foi desenvolvido um sistema para visualização do campo de batalha chamado *Dragon System* (Durbin et al., 1998). Este sistema roda em uma mesa de trabalho de realidade virtual. O sistema mostra uma representação tridimensional do campo de batalha que inclui um mapa do terreno, entidades representando unidades amigas, inimigas, desconhecidas e neutras, além de simbologias representando outras características tais como obstáculos e pontos chave para o plano de operações (Figura 2.3). Pode-se observar que algumas unidades são caracterizadas com as simbologias tradicionais no formato padronizado em uso pelas forças armadas adaptados ao ambiente 3D, ao mesmo tempo em que se misturam modelos 3D como, por exemplo, o carro de combate representando um regimento blindado (Figura 2.3-b).

(a) (b)
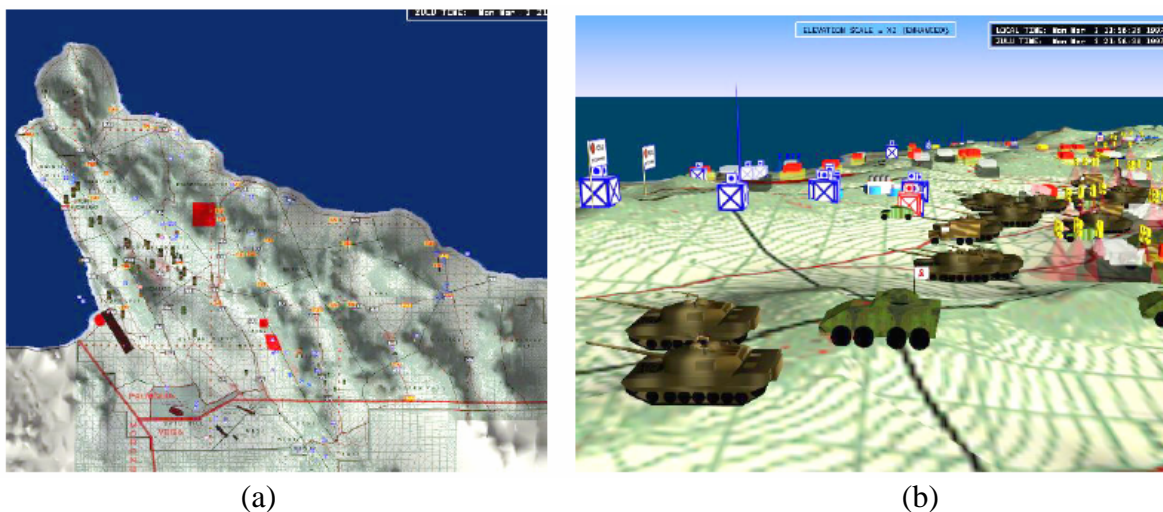
Figura 2.3 – Sistema Dragon: (a) Vista geral do mapa;
(b) Entidades representadas por modelos e símbolos.

Considerado como sendo o principal sistema digital de Comando e Controle do Exército dos EUA no nível Brigada, o *Force XXI Battle Command Brigade and Bellow* (FBCB2) fornece a situação operacional no nível tático e permite um fluxo rápido e eficiente de informações no campo de batalha (Kauchak, 2001). Porém, este sistema trabalha apenas em duas dimensões utilizando mapas cartográficos no formato raster (Figura 2.4).

Figura 2.4 – Sistema FBCB2 mostrando mapa em 2D.

O *Joint Operations Visualization Environment* (JOVE) foi desenvolvido para auxiliar os tomadores de decisão militares. O sistema otimiza a exibição de dados táticos para a apresentação de informações em três telas de projeção controladas por um computador Onyx 2 Infinite Reality 2 da SGI (Feibush, Gagvani e Williams, 2000). O software é baseado no paradigma cliente-servidor sendo que os diferentes módulos comunicam-se através de CORBA (*Common Object Request Broker Architecture*).
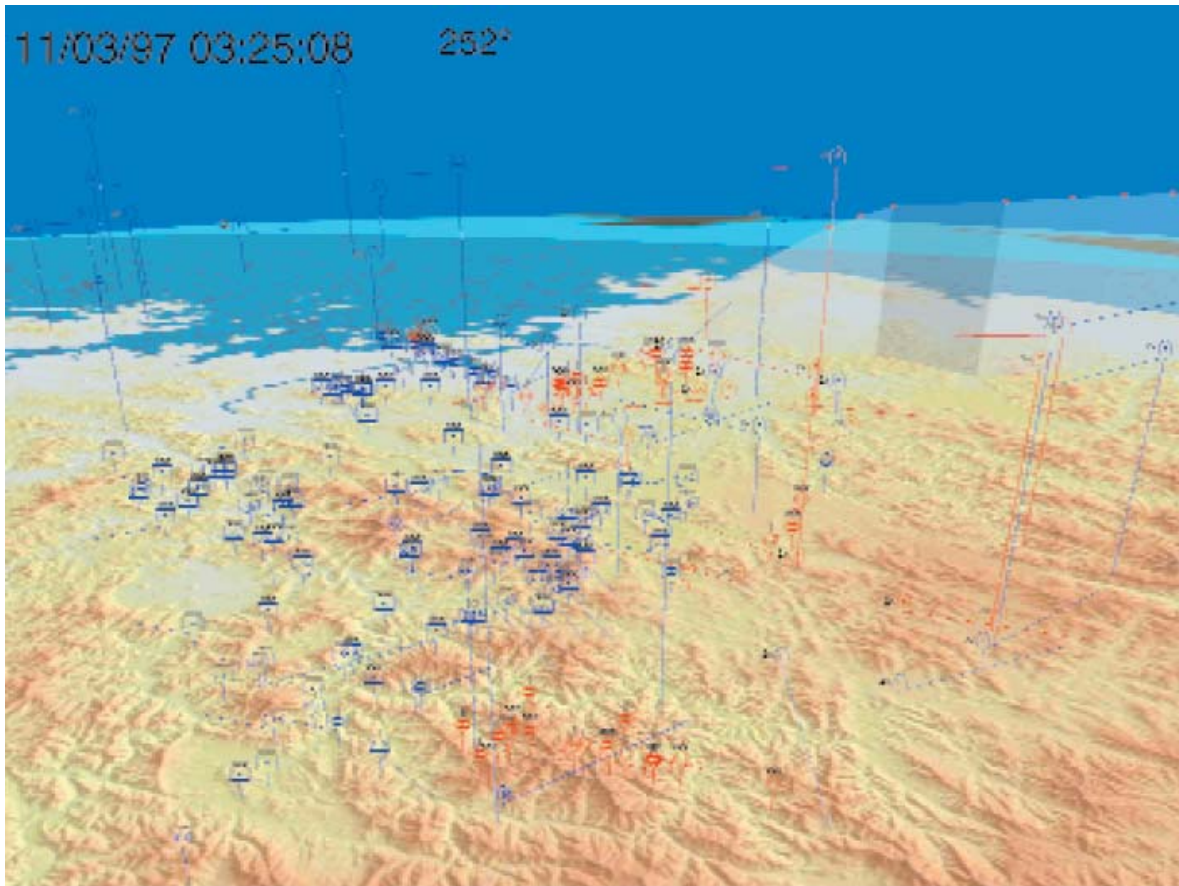
Figura 2.5 – JOVE: unidades militares representadas simbolicamente.

A empresa italiana especializada em sistemas de defesa *Marconi Communications* comercializa produtos para simulação e treinamento e sistemas de comando e controle, dentre eles o sistema denominado MACCIS (*Marconi Automated Command and Control Information System*), baseado na doutrina de emprego do exército italiano. O módulo de visualização de cartas também trabalha com cartas raster de forma bidimensional e emprega a representação dos objetos utilizando somente uma simbologia em duas dimensões (Figura 2.6).
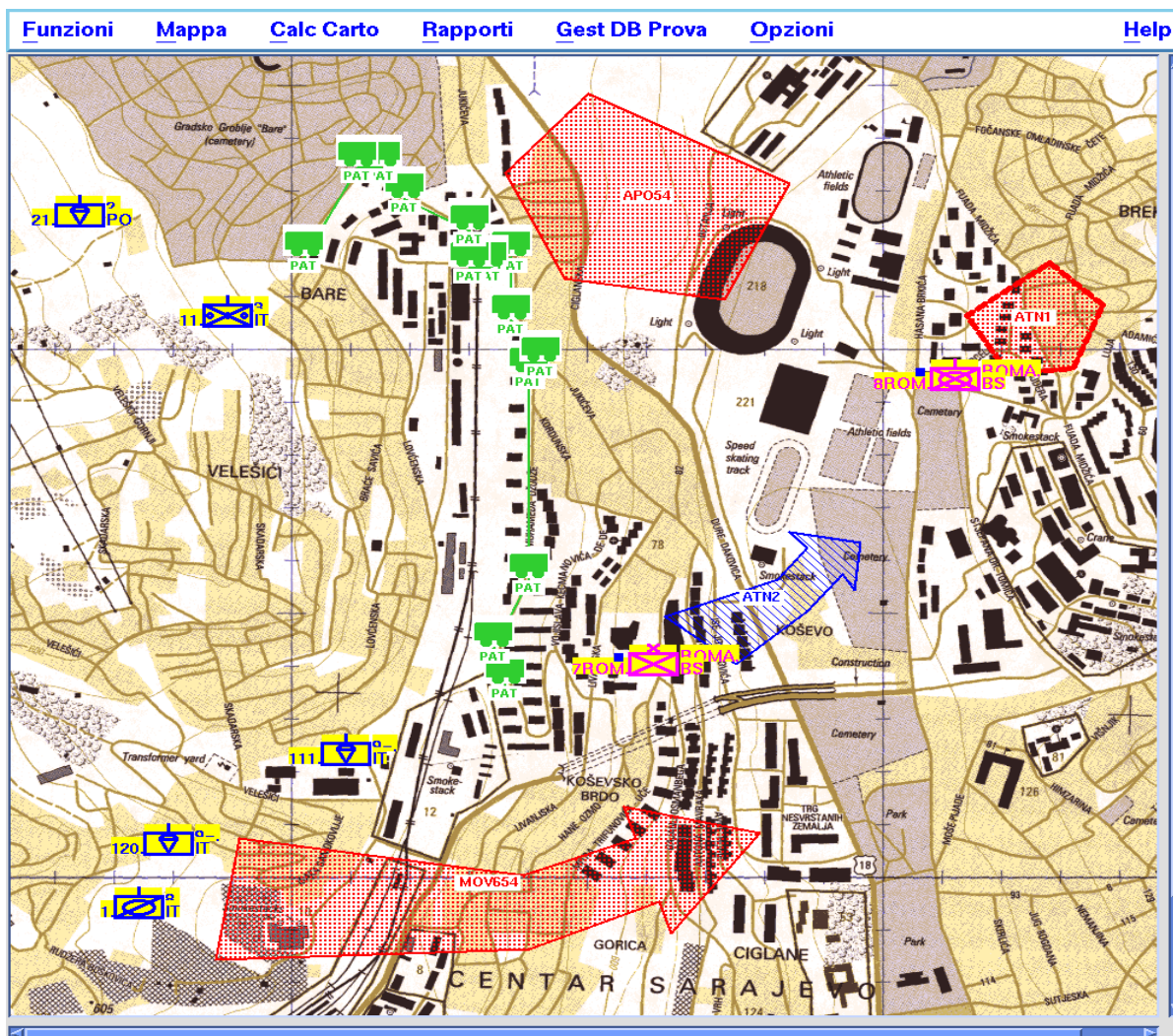
Figura 2.6 – MACCIS: representação por símbolos sobre carta ao fundo.

No Brasil as iniciativas na área de comando e controle encontram-se ainda bastante tímidas. Pode-se citar o trabalho realizado por Pierre (2002) onde foi desenvolvida uma aplicação para visualização de mapas em assistentes pessoais (PDA) acoplados a equipamentos de posicionamento por satélite (GPS) para emprego por tropas de grande mobilidade.

Outro trabalho brasileiro foi o realizado por Mello (2003) que descreve o desenvolvimento do sistema batizado de Tesamará. Este sistema é uma ferramenta de apoio à tomada de decisões em procedimentos de combate e possui como principal funcionalidade a visualização tridimensional do campo de batalha (Figura 2.7) com a vantagem de utilizar as simbologias empregadas pelas Forças Armadas Brasileiras.
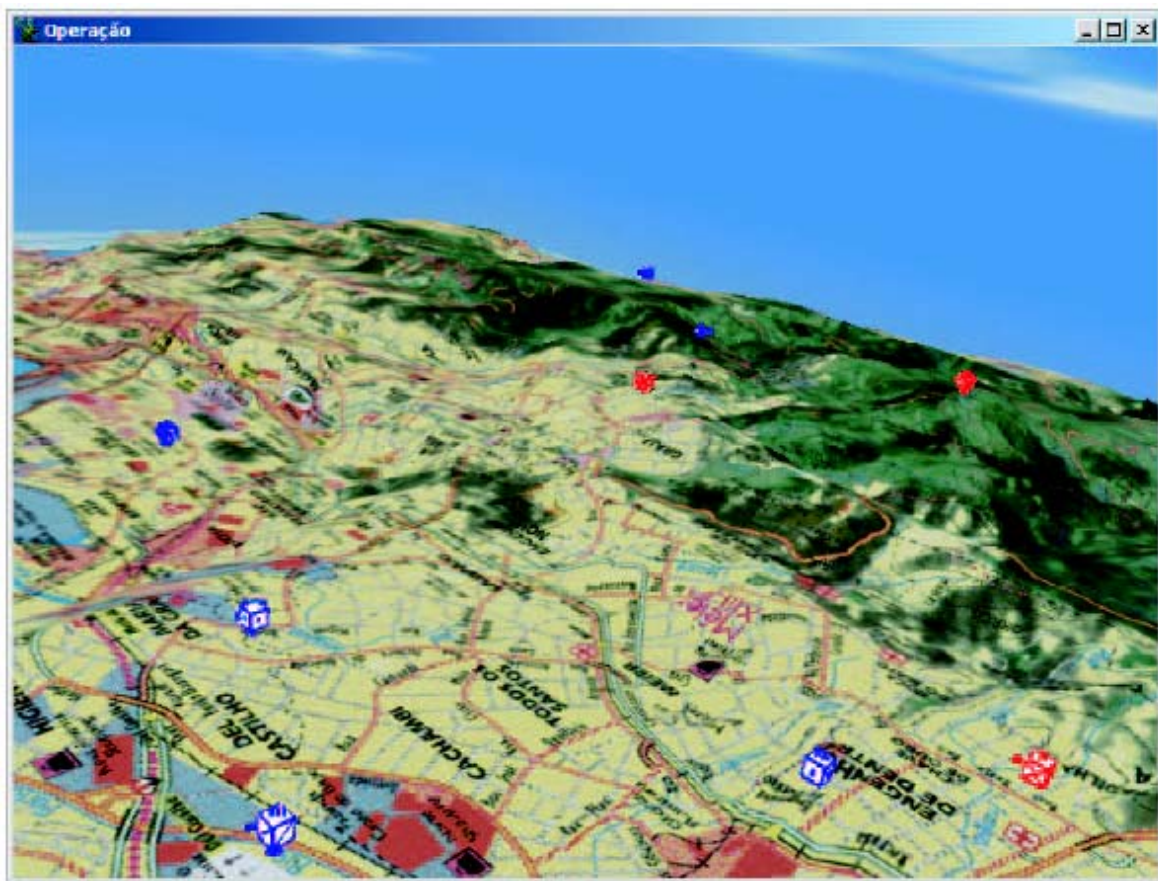
Figura 2.7 – Tesamará: visualização de uma operação.

CAPÍTULO 3

MODELAGEM E RENDERIZAÇÃO DE TERRENOS

Nos diversos tipos de ambientes virtuais os modelos digitais de terrenos são a ferramenta principal para a apresentação e a comunicação de informação espacial.

A modelagem de terreno é o processo de quantificar superfícies Este campo vem sofrendo grande revolução desde o final do século 20 com a manipulação de matrizes espaciais contendo uma amostragem regular da elevação de terrenos mais conhecidos como Modelos Digitais de Elevação (MDE), do inglês *Digital Elevation Models* - DEM, os quais quantificam e retratam a superfície de uma área (Figura 3.1-a).

A visualização do terreno é um importante componente em muitas aplicações civis e militares. O conjunto de entrada de um sistema de visualização de terreno geralmente é um grande Modelo Digital do Terreno (MDT), que consiste de um conjunto de dados altimétricos amostrados em uma grade regular, ou seja o MDE, e o conjunto correspondente de dados de texturas associadas como, por exemplo, fotos de satélite ou aéreas, que são mapeadas na superfície de terreno reconstruída. Sendo assim, a chave para uma renderização eficiente do terreno está na manipulação eficiente dos dados geométricos e de textura, principalmente quando a base de dados possui várias ordens de magnitude superior ao tamanho da memória do sistema (Rabinovich e Gotsman, 1997).

3.1 – CONCEITOS

Em computação gráfica um terreno é usualmente representado por malhas trianguladas (*triangle meshes*), que são conjuntos de triângulos interconectados onde não mais de dois triângulos compartilham um mesmo lado. Essas triangulações podem se basear em grades regulares (Figura 3.1-b) ou em redes de triangulações irregulares, mais conhecidas como TIN (*triangulated irregular networks*) (Figura 3.1-c). Como essas representações de terrenos são tipicamente compostas por milhões de triângulos, os hardwares atuais não possuem a capacidade de renderizar essas grandes malhas de forma interativa a altas taxas de quadros (*frame rates*) (Youbing *et al.*, 2001). Portanto é necessária a utilização de metodologias que reduzam a quantidade de triângulos a serem renderizados sem que haja significante deterioração na visualização.

(a) (b) (c)
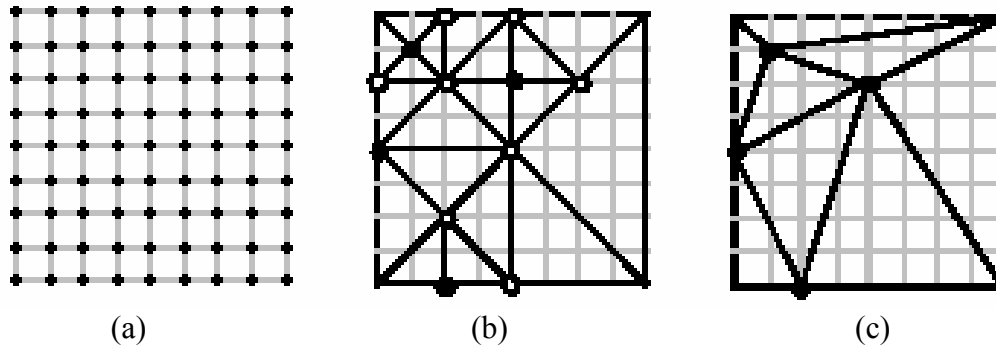
Figura 3.1 - (a) MDE; (b) Triangulação baseado em uma grade regular; (c) TIN

Uma representação de terreno baseado em um MDE típico consiste de um número muito grande de polígonos de forma que, mesmo hardwares gráficos de alta performance podem ter dificuldade para mostrar taxas de quadros razoáveis. Para diminuir a complexidade da cena mantendo a qualidade da imagem, os algoritmos que realizam a triangulação do terreno levam em conta as seguintes propriedades (Röttger *et al.*, 1998):

- Considerando o quanto é acidentado o terreno: áreas planas ou suaves podem ser aproximadas por menos triângulos do que regiões mais acidentadas;
- Considerando o ponto de vista do observador: as regiões mais próximas devem ser aproximadas com maior precisão do que as mais distantes.

Para superar as limitações impostas pelo hardware e explorar as propriedades anteriormente descritas, as abordagens geralmente utilizadas para reduzir a quantidade de geometrias a serem renderizadas foram designadas como algoritmos de níveis de detalhes, ou LOD (*Level of Detail*). Uma estrutura de dados de um MDE que possibilita a extração de uma superfície triangulada com uma LOD arbitrária é chamada de modelo de triangulação em multiresolução.

Existem os métodos de redução na quantidade de triângulos a serem renderizados baseados em um conjunto de níveis de detalhes discretos, que representam o mesmo objeto com uma quantidade diferente de triângulos. A determinação do nível de detalhe a ser utilizado pode ser definido com base na distância entre o observador e o objeto ou no erro da representação do objeto no espaço de tela.

Métodos de níveis de detalhes discretos vão apresentar dois problemas. O primeiro será que grandes objetos, onde terreno é um caso típico, vão possuir regiões próximas ao observador e outras mais distantes que deverão ser renderizadas em níveis de detalhes diferentes. Em segundo, a mudança de um nível de detalhe para outro pode levar a artefatos temporais de *aliasing* mais conhecidos como *popping*. Para resolver o problema do *popping* foram propostos alguns métodos:

- Utilizar uma grande quantidade de níveis de detalhes para reduzir a diferença entre dois níveis consecutivos, porém resultando em um grande espaço de armazenagem;

- Utilizar modelos muito complexos com erros no espaço de tela pequenos, o que vai consumir grande poder de renderização;
- Realizar um processo de transformação (*morph*) de um nível de detalhe para outro, através de uma animação das posições dos vértices ou misturar (*blend*) no novo nível de detalhe o resultado de vários quadros (*frames*), o que também é um método computacionalmente pesado.

Desta forma, métodos baseados em níveis de detalhes discretos são inadequados para a visualização de terrenos. Sendo assim, para lidar com os problemas apresentados anteriormente foram idealizados algoritmos que gerenciam níveis de detalhes contínuos (*continuous level of detail*) computando o nível de detalhe apropriado de cada triângulo em cada quadro, muitas vezes aproveitando a coerência entre quadros para minimizar a diferença na triangulação entre quadros (Ögren, 2000). Lindstrom *et al.* (1996) fornece uma explicação melhor sobre a definição de níveis de detalhes contínuos.

Heckbert e Garland (1997) categorizaram os métodos para a simplificação e aproximação de superfícies poligonais para o caso de terrenos em seis classes:

- Grades regulares (*Regular grids methods*): são as técnicas mais simples. Utilizam uma grade de amostras igualmente e periodicamente espaçadas. Grades regulares também são conhecidas como grades uniformes ou como modelo digital de elevação (MDE);
- Subdivisão hierárquica (*Hierarchical subdivision methods*): são baseadas em quadtrees, kd-tree e outras triangulações hierárquicas que utilizem a estratégia de dividir para conquistar, dividindo o terreno em pequenas regiões de forma recursiva para construir uma árvore de regiões. São métodos geralmente rápidos, simples e que facilitam a modelagem em multiresolução. Em perspectivas, onde as porções mais próximas do terreno requerem maior detalhamento do que as regiões mais distantes, a hierarquia facilita a renderização com a adaptação dos níveis de detalhes;
- Por feições (*Feature methods*): baseia-se na seleção dos conjuntos de pontos que compõem características ou feições importantes do terreno para realizar a triangulação. São considerados pontos importantes, feições ou pontos críticos e seus limites elementos como feições topográficas como, por exemplo, picos, buracos, cordilheiras e vales. São métodos que despertam interesse na cartografia;
- Refinamento (*Refinement methods*): são algoritmos que realizam múltiplas passagens começando com uma aproximação inicial mínima e, em cada passagem, um ou mais vértices são inseridos na triangulação. Este processo é repetido até se chegar a um erro ou a um número de vértices desejados.
- Decimação (*Decimation methods*): inicia com a triangulação de todos os pontos de entrada e, iterativamente, a cada passagem são retirados vértices, triângulos ou outras feições geométricas da triangulação gradualmente simplificando a aproximação;
- Ótimo (*Optimal methods*): aqueles que foram incluídos apenas por suas propriedades teóricas.

Outra revisão pode ser encontrada no trabalho de Cignoni, Montani e Scopigno (1998), que fornece uma classificação e algumas comparações de métodos e abordagens para a simplificação de superfícies. Oferece também uma avaliação sobre os erros de aproximação que são introduzidos durante o processo de simplificação.

A maior parte dos modelos em multiresolução suportam um tipo específico de dados de entrada: podem ser pontos distribuídos arbitrariamente para as redes de triangulação irregulares (*Triangulated Irregular Networks – TIN*) ou pontos regularmente distribuídos para as grades regulares.

3.2 – CAMPOS DE ALTURAS E GRADES REGULARES

Os campos de alturas (*height fields*) constituem-se de uma grade regular cujas amostras de altimetria do terreno estão igualmente espaçadas.

Uma abordagem comum nos algoritmos para renderização de terrenos baseados em campos de alturas é a divisão do terreno em áreas quadradas conhecidas como ladrilhamento (*tiles*) (Figura 3.2-a), nos quais ficam armazenados os diferentes níveis de detalhes. Os ladrilhamentos mais próximos do observador são renderizados com um nível de detalhes maior que aqueles mais distantes.

Esta abordagem pode introduzir problemas na fronteira entre ladrilhos. As transições entre níveis de detalhes podem ser percebidas pelo observador como também podem introduzir falhas e descontinuidades espaciais na malha mais conhecidas como *cracks* (Figura 3.2-b).

(a) (b)
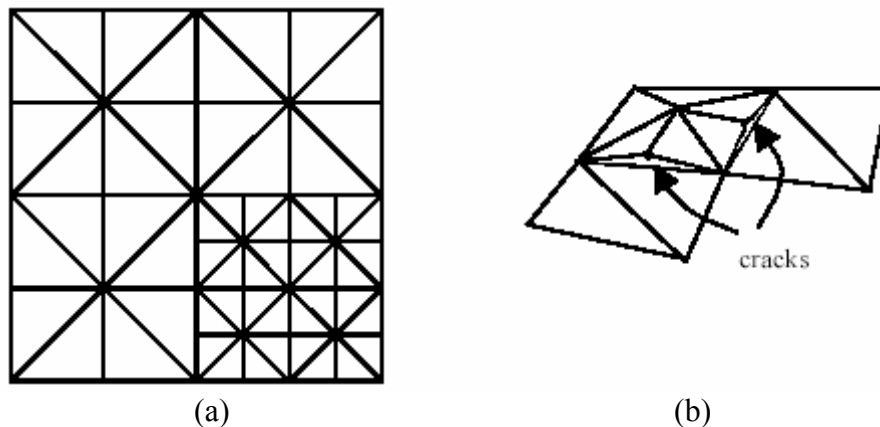
Figura 3.2 – (a) Subdivisão em *tiles*; (b) *Cracks*.

Esta descontinuidade pode ser evitada quando se assegura de que a projeção no plano *xy* do lado de um triângulo não contenha vértices de outros triângulos. Um método muito utilizado é a costura dos ladrilhos como mostrado na Figura 3.3. Este tipo de solução vai reduzir as diferenças visuais entre níveis de detalhes diferentes.
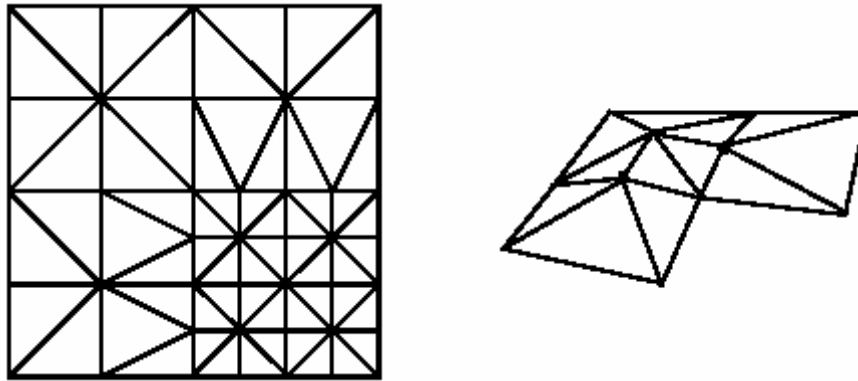
Figura 3.3 – Ladrilhamento (subdivisão em *tiles*) com costuras para eliminar *cracks*.

Na simplificação de campos de alturas e de superfícies paramétricas um dos métodos mais empregados é o de subdivisão hierárquica, por serem geralmente rápidos, simples e por facilitarem a criação de modelos em multiresolução. A contrapartida pela simplicidade e rapidez está em aproximações de qualidade inferior o outros métodos mais generalizados de triangulação.

3.3 – TRIANGULAÇÕES BASEADAS EM QUADTREES

A quadtree é uma das estruturas de dados espaciais hierárquicas considerada como das mais eficientes para representar dados de grades regulares, como os campos de alturas, em um modelo de terreno em multiresolução. A Figura 3.4 mostra um exemplo de simplificação de triangulação de terreno utilizando um método de multiresolução baseada em quadtree.
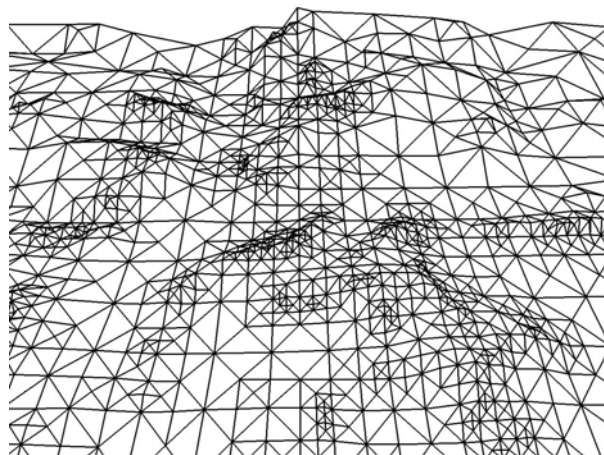
Figura 3.4 – Triangulação de terreno baseada em quadtree.

A característica de uma quadtree é a de possuir quatro filhos para cada nó da árvore. No contexto da renderização de terrenos, o nó raiz representa a região quadrada que envolve todo o campo de alturas. Cada filho representa os quatro quadrantes que compõem o nó raiz e cada um dos filhos é recursivamente dividido em conjuntos de quatro quadrantes até uma resolução determinada (Figura 3.5).
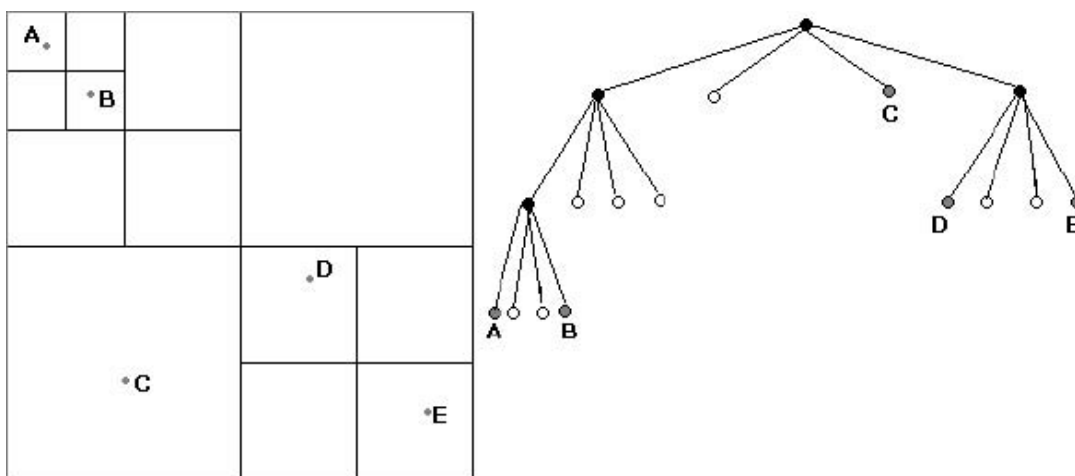
Figura 3.5 – Representação do espaço por uma quadtree.

A utilização de quadtrees é motivada pela simplicidade em sua implementação e também por sua eficiência nos mecanismos para a realização do corte do volume de visualização.

Existem diversas abordagens que utilizam a triangulação de campos de alturas em multiresolução baseados em quadtrees. Pajarola (2002) apresenta e compara a eficiência de algumas das metodologias para a triangulação e visualização de terrenos baseados em quadtrees de acordo com as seguintes características:

- Rapidez na adaptação da triangulação;
- LOD contínuo;
- Eficiência na métrica de erro geométrico;
- Performance na renderização das geometrias;
- Eficiência no acesso espacial;
- Compactação na armazenagem e na representação;

Como exemplos de sistemas baseados em quadtrees, dois trabalhos de visualização de terrenos com níveis de detalhes contínuos serão resumidamente apresentados a seguir.

3.3.1 – RENDERIZAÇÃO COM LOD CONTÍNUO

Durante o SIGGRAPH de 1996, Lindstrom *et al.* (1996) propuseram um dos primeiros algoritmos para a renderização de terrenos em tempo real com níveis de detalhes contínuo.

O algoritmo proposto trabalha por blocos e utiliza uma estratégia *bottom-up*. O processo inicia-se com uma triangulação grosseira de toda a grade do conjunto de dados do terreno para determinar quais os níveis de detalhes serão necessários. A seguir a malha da triangulação vai sendo recursivamente simplificada através da união de triângulos. Ambos os passos são executados para cada quadro sendo que, as avaliações envolvidas nas simplificações, são realizadas em tempo real e são baseadas no ponto de vista do observador e na geometria do terreno.

Uma malha primitiva é a menor representação por uma malha de triangulação simétrica e possui dimensões 3 x 3 vértices (Figura 3.6-a). Malhas maiores são formadas por malhas menores agrupadas em conjuntos de 2 x 2 (Figura 3.6-c).

(a) (b) (c)
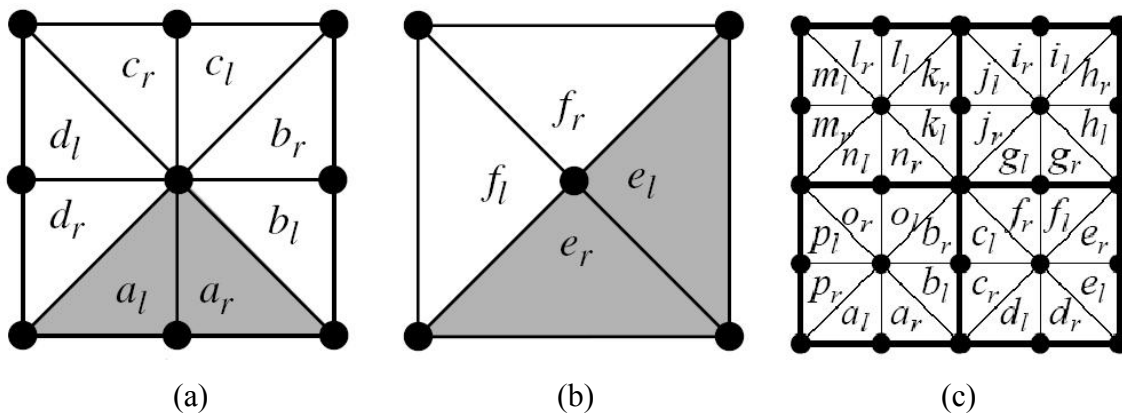
Figura 3.6 – (a) Malha primitiva; (b) unindo triângulos ao longo da diagonal; (c) bloco com 2x2 malhas primitivas

A união de triângulos dentro das malhas primitivas é realizada em duas fases. Na primeira, os pares de triângulos isósceles que compartilham um dos catetos são fundidos e o ponto no meio da lateral do quadrado é removido ($\boldsymbol{a_l}$ e $\boldsymbol{a_r}$ da Figura 3.6-a). Na segunda fase, o vértice central da região quadrada é removido com a união dos pares de triângulos isósceles cujos catetos estão sobre a diagonal do quadrado ($\boldsymbol{e_l}$ e $\boldsymbol{e_r}$ da Figura 3.6-b). Para prevenir *cracks* sempre dois pares de triângulos isósceles que compartilham o mesmo vértice a ser removido devem ser unidos ao mesmo tempo ($\boldsymbol{e_l}$, $\boldsymbol{e_r}$ e $\boldsymbol{f_l}$, $\boldsymbol{f_r}$ da Figura 3.6-b).

Para que os pares de triângulos sejam reduzidos a um único triângulo e que, por sua vez, seja considerado para simplificações posteriores de forma recursiva, certos critérios devem ser obedecidos. Um desses critérios é definido pelo grau de variação na inclinação entre os dois triângulos a serem reunidos. Seja a Figura 3.7: para os triângulos ΔABE e ΔBCE, com A, B e C no plano perpendicular ao plano *xy*, a variação da inclinação é medido pela distância vertical (eixo z):

$$\delta_B = \left| B_z - \frac{A_z + C_z}{2} \right|$$

Esta distância é chamada valor delta do vértice B. Quanto maior o valor de delta, menor as chances de se realizar uma união de triângulos.
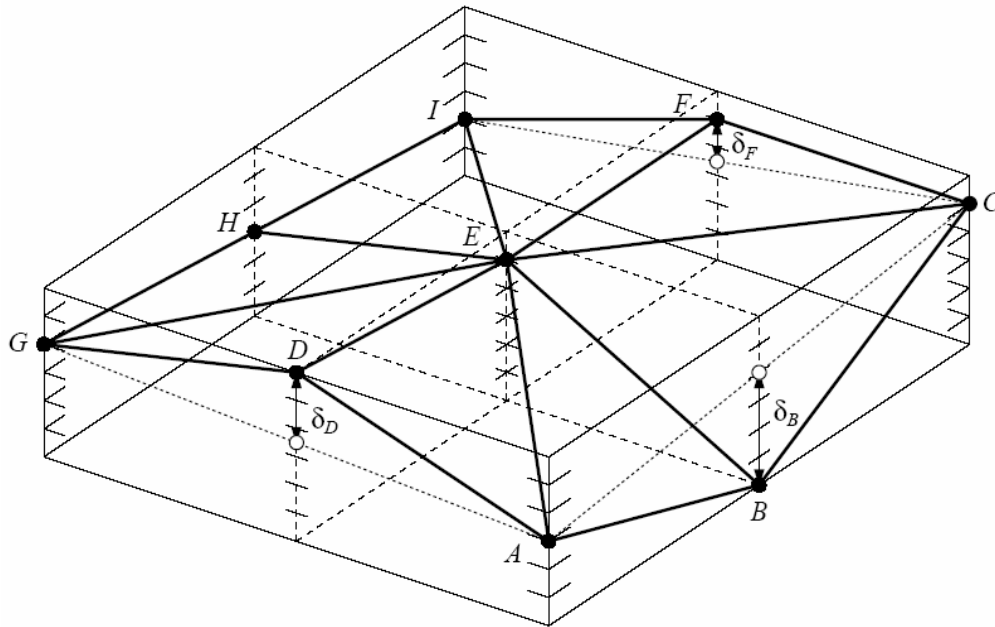
Figura 3.7 – Representação gráfica dos valores delta.

Para uma simplificação mais grosseira, o algoritmo trabalha por blocos e faz a seleção do nível de detalhe apropriado com base nos valores delta dos vértices que

compõem um bloco encontrados pela metodologia anteriormente apresentada. Se o maior valor de delta de todos os vértices em um bloco for menor que um limite pré-estabelecido, então esses vértices podem ser descartados e o bloco pode ser substituído por outro de menor resolução que, por sua vez, é considerado posteriormente para outras simplificações.

Uma das grandes contribuições deste trabalho foi a definição de uma métrica de erro eficiente dependente do ponto de vista do observador e que se provou ser uma forma eficaz de se avaliar o grau de importância dos vértices a serem selecionados. Após a perspectiva o valor delta é projetado para o espaço da tela: se o valor encontrado for menor que uma tolerância pré-determinada o vértice pode ser removido e o par de triângulos unidos (Figura 3.8).
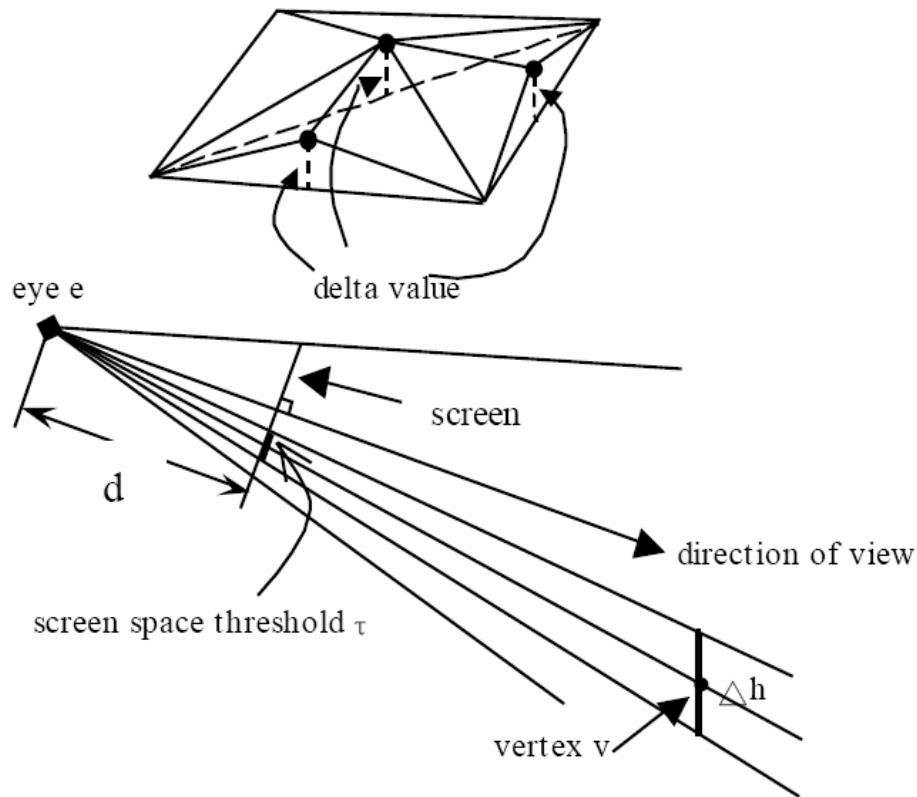
Figura 3.8 – Métrica de erro no espaço da tela.

A fim de eliminar *cracks* entre nós adjacentes em diferentes níveis, utilizou-se uma árvore binária de vértices para manter a relação entre vértices. Blocos adjacentes devem compartilhar vértices em suas fronteiras para evitar *cracks* entre blocos.

O algoritmo aproveita a coerência que existe entre quadros consecutivos para reduzir o número de vértices processados por quadro. As malhas resultantes são geradas a partir de uma subdivisão do espaço por uma quadtree.

3.3.2 – LOD CONTÍNUO PARA CAMPOS DE ALTURAS

O trabalho de Röttger *et al.* (1998) propôs um método para a eliminação de *cracks* que vem sendo muito utilizado desde então. O algoritmo também é baseado em grades regulares e utiliza uma estratégia *top-down* para criar a triangulação.

A estrutura de dados utilizada baseia-se em uma quadtree para permitir uma operação de recorte (*clipping*) eficiente. A triangulação é realizada, recursivamente, pela varredura descendente da quadtree: quando se chega a uma folha da quadtree, uma *triangle fan* parcial ou total é desenhada. Para evitar as descontinuidades entre blocos adjacentes de resoluções diferentes, a malha gerada simplesmente não considera o ponto no meio do lado comum, como pode ser visto no exemplo da Figura 3.9.
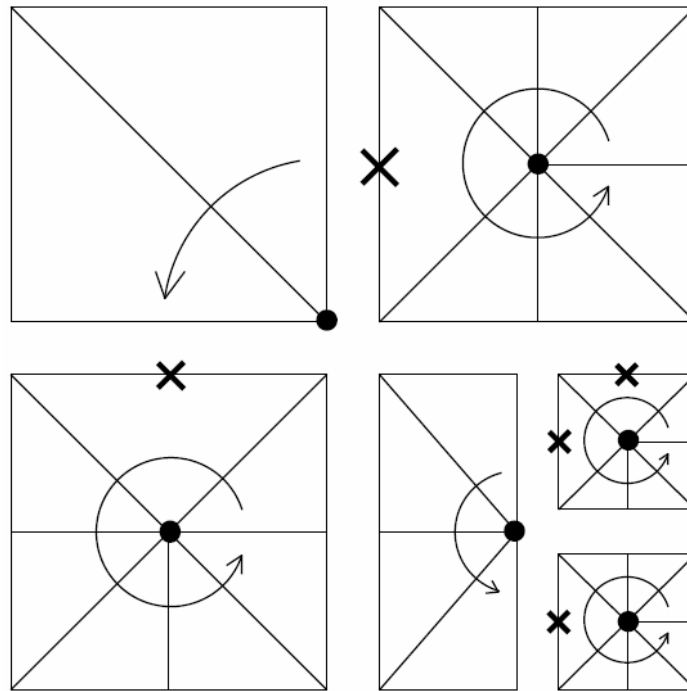
Figura 3.9 – Geração recursiva da triangulação de uma *height field* de 9x9.
Os **x** indicam vértices que foram desconsiderados.

Em cada nó da quadtree, um critério é avaliado para determinar se chegou ao nível de detalhe mais fino. O algoritmo e utiliza como critério um híbrido de distância do ponto de vista do observador com o quão acidentado é a superfície local.

Desta forma, considera-se que a resolução diminui a medida que a distância do observador aumenta. Esta condição pode ser garantida por:

$$\frac{l}{d} < C$$

onde l é a distância do ponto de vista do observador, d é o comprimento do bloco considerado e C é uma constante que serve como parâmetro qualitativo e controla a resolução mínima global (Figura 3.10).
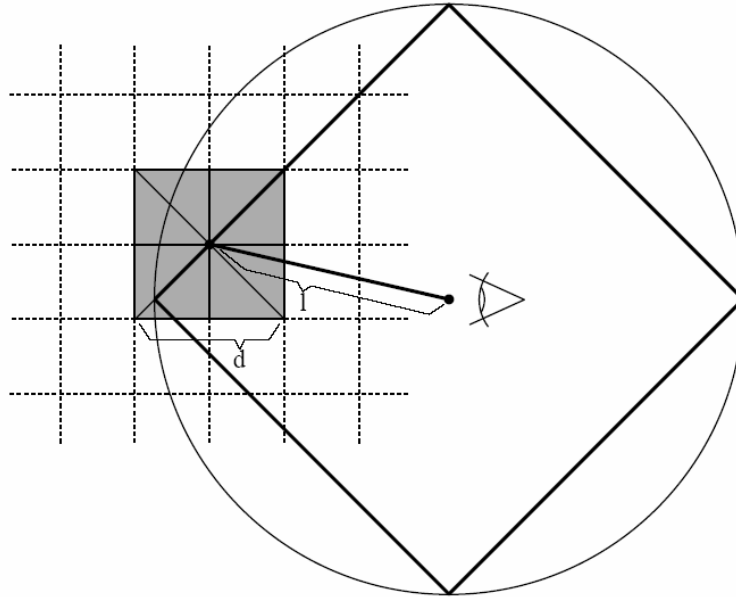
Figura 3.10 – Critério de distância do ponto de vista do observador.

O outro critério considera que deseja-se aumentar a resolução para as regiões mais acidentadas. Quando se cai um nível na hierarquia, novos erros são introduzidos em cinco pontos: no centro e nos quatro pontos médios dos lados. Um limite superior para a aproximação do erro no espaço tridimensional pode ser dado pelo máximo dos valores absolutos das diferenças de alturas calculados nos quatro lados do bloco e ao longo das duas diagonais (Figura 3.11).
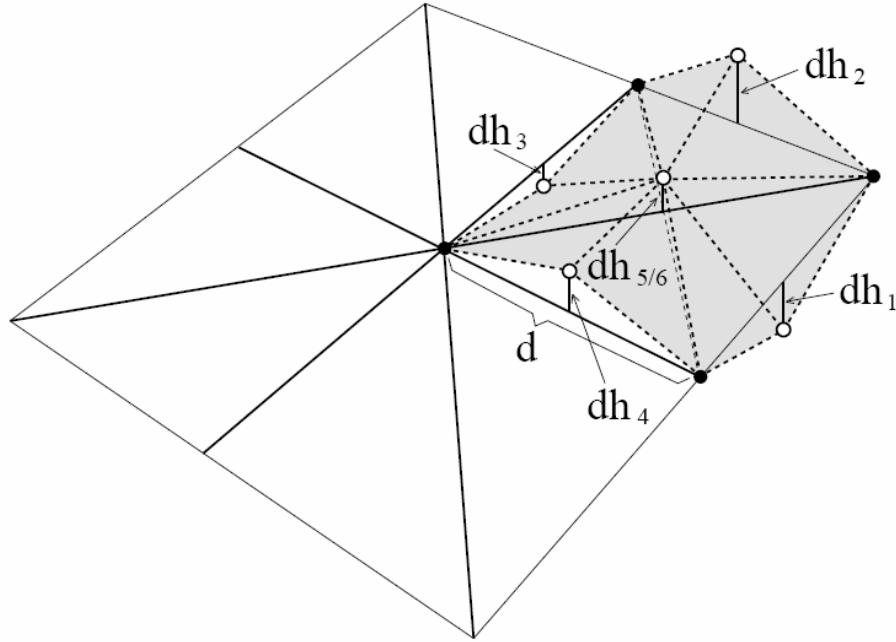
Figura 3.11 – Medindo erros devido às irregularidades da superfície do terreno.

O erro introduzido ao diminuir um nível na quadtree pode ser computado pelo pré-cálculo do máximo dos valores absolutos das diferenças de elevação que indicam o quanto é acidentado o terreno, e que foi definido como:

$$d2 = \frac{1}{d} \max_{i=1..6} |dh_i|$$

Revisando o primeiro critério, pode-se definir um parâmetro *f* dado por:

$$f = \frac{l}{d \cdot C \cdot \max|c \cdot d2,1|}$$

onde, se $f < 1$, o nó vai ser subdividido. A constante *c* especifica a resolução global desejada e vai influenciar o número de polígonos renderizados a cada quadro.

Como existe uma dependência entre nós adjacentes na quadtree, o cálculo do erro garante que a máxima diferença entre níveis de nós adjacentes nunca será maior que 1, desta forma reduzindo o trabalho para a eliminação de *cracks*. Além disso, o algoritmo realiza geomorfismos, o que reduz ou mesmo elimina o efeito de *popping*.

3.4 – ROAM

Uma das mais conhecidas metodologias para simplificação de malhas em grades regulares foi proposta por Duchaineau *et al.* (1997) e que foi denominada ROAM (*Real-time Optimally Adapting Meshes*) que, desde então, vem sendo muito utilizada para simplificação de terrenos principalmente em jogos.

O ROAM realiza a divisão ou a união no que foi definido como diamantes. O algoritmo utiliza uma estrutura de árvore binária de triângulos com filas de prioridades para a divisão (*split*) e a união (*merge*) de diamantes para realizar atualizações progressivas na malha de triângulos (Figura 3.12). Em ambas as filas os diamantes são ordenados por suas prioridades, determinadas pelos valores do erro de projeção na tela ou outras métricas de erro. De modo geral, quatro fases são computadas por quadro:

1. Atualização recursiva e incremental do recorte do volume de visualização (*view-frustum culling)*;
2. Atualização das prioridades dos triângulos que potencialmente possam ser unidos ou divididos na fase 3;
3. Atualização da triangulação utilizando divisões e uniões definidos pelas filas de prioridades;
4. Atualizações necessárias nas *triangle strips* para as alterações impostas pelas fases 1 e 3;
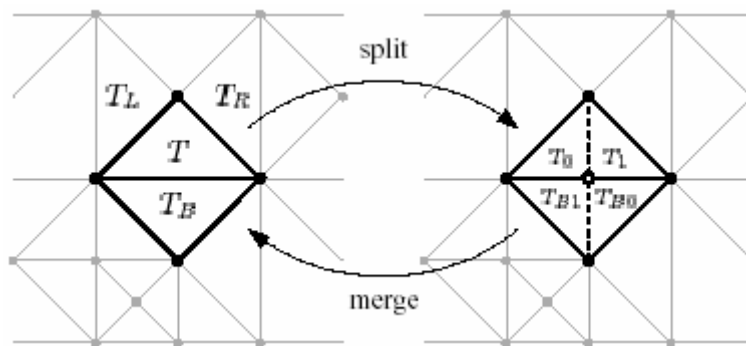
Figura 3.12 – Um diamante e as operações de divisão e união de triângulos.

O algoritmo ROAM fundamenta-se na estrutura de dados que foi batizada de árvore binária de triângulos (*binary triangle tree*) onde pode-se observar uma analogia a uma quadtree. A Figura 3.13 mostra os primeiros níveis de uma árvore binária de triângulos.
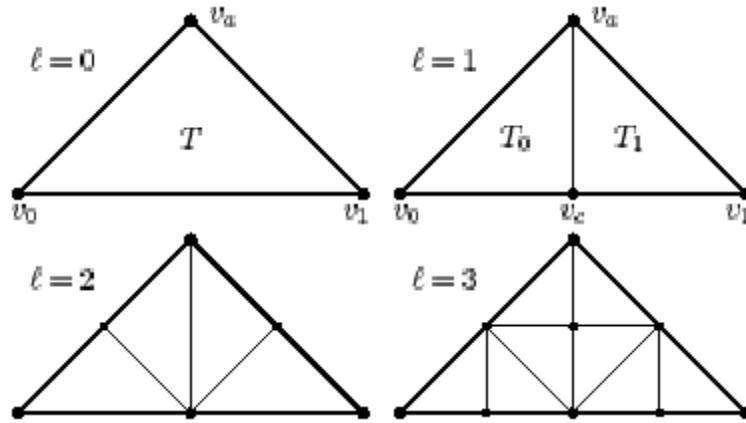
Figura 3.13 – Níveis de 0 a 3 de uma árvore binária de triângulos.

Para evitar descontinuidades na malha, um triângulo não pode ser simplesmente dividido se o triângulo que está em sua base na malha pertencer a um nível mais baixo. Para tanto serão necessárias uma série de divisões forçadas em uma seqüência recursiva, como no exemplo mostrado na Figura 3.14.
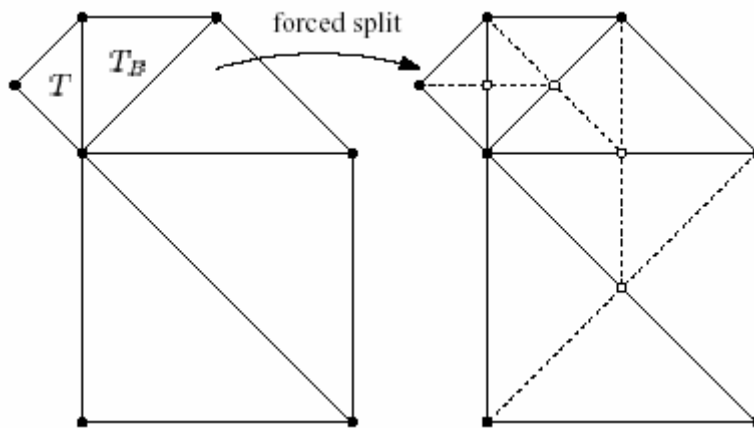
Figura 3.14 – Divisões forçadas para que o triângulo **T** seja divido uma vez que $\mathbf{T_B}$ é de um nível mais baixo.

Turner (2000) apresenta uma explicação do algoritmo ROAM assim como uma orientação para sua implementação.

3.5 – TIN

Uma TIN (*Triangular Irregular Network*) é um conjunto de triângulos adjacentes e que não se sobrepõe, cujos vértices são adaptativamente definidos a partir de um MDE (Figura 3.15). A geração automática de modelos de TIN a partir de MDE é uma área de grande interesse para a pesquisa.
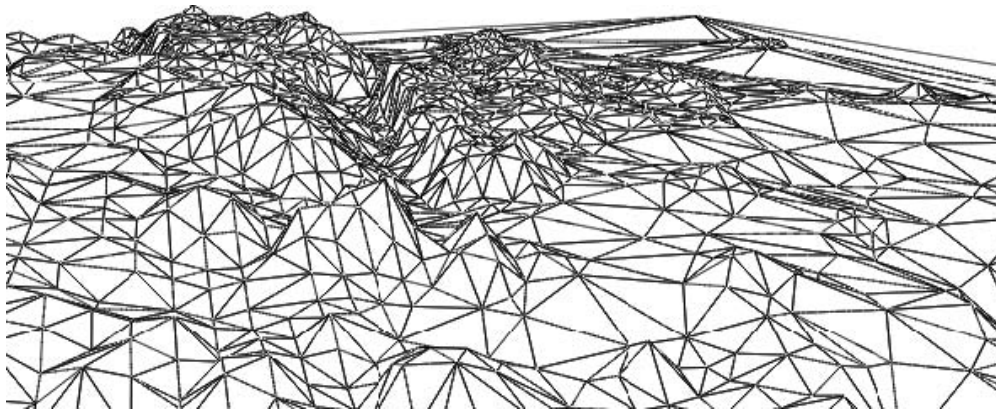
Figura 3.15 – *Triangular Irregular Network* (TIN)

Em seu trabalho, Lee (1991) discutiu e comparou quatro métodos de simplificação de superfícies que tinham como resultado a geração de TINs a partir de MDEs. Cada superfície aproximada por uma TIN era comparada com a superfície do MDE original pela diferença de elevação em cada ponto do MDE, sendo calculados médias e desvios padrões dessas diferenças. Além disso, utilizando o índice de Moran, um coeficiente de autocorrelação espacial, era realizada a análise da distribuição espacial dessas diferenças.

Como representativo exemplo da utilização de um TIN para a representação em multiresolução de uma superfície a partir de um MDE, será apresentado resumidamente a seguir o trabalho de Hoppe (1997, 1998).

3.5.1 – VDPM

Em 1998 Hoppe apresentou uma metodologia que permitiu o sobrevôo em tempo real de um grande terreno. Esta metodologia foi chamada *View Dependent Progressive Meshes* (VDPM). O terreno é dividido em blocos e, para cada bloco, é gerada uma TIN de acordo com o ponto de vista do observador e a métricas de erro do espaço de tela.

Inicialmente Hoppe (1996) introduziu uma representação que define uma malha arbitrária de triângulos como uma seqüência de aproximações de malhas otimizadas para nível de detalhe independente do observador (*view-independent LOD)*, o qual definiu como *progressive meshes* (PM). Uma malha arbitraria $\hat{M}$ é armazenada como sendo uma malha grosseira (mais simplificada), M^0, junto com uma seqüência de *n* registros detalhados que indicam como refinar incrementalmente M^0 para a malha original $\hat{M} = M^n$.

Para construir uma representação através de PM, uma malha arbitraria $\hat{M}$ é simplificada através de uma seqüência de *n* transformações denominadas *edge collapse* (*ecol* na Figura 3.16) para se chegar a uma malha mais simplificada M^0:

$$(\hat{M} = M^n) \xrightarrow{ecol_{n-1}} \cdots \xrightarrow{ecol_1} M^1 \xrightarrow{ecol_0} M^0$$

A seqüência de transformações *ecol* é definida por um processo de otimização que busca preservar a aparência do modelo. A transformação inversa do *ecol* foi chamada de *vertex split*, uma transformação elementar que adiciona um vértice à malha (Figura 3.16) e, assim, o processo pode ser revertido:

$$M^0 \xrightarrow{vsplit_0} M^1 \xrightarrow{vsplit_1} \cdots \xrightarrow{vsplit_{n-1}} (M^n = \hat{M})$$

O conjunto (M^0, $\{vsplit_0, \ldots, vsplit_{n-1}\}$) forma uma representação por PM de $\hat{M}$.
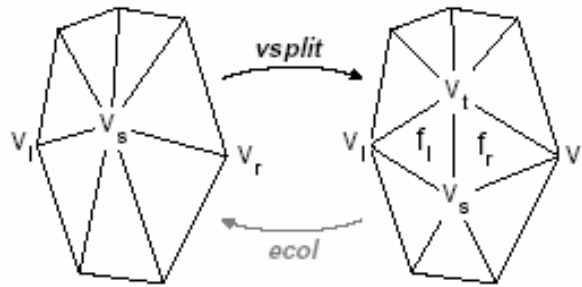
Figura 3.16 – Transformações: *vertex split* e *edge collapse*

Como mostrado por Hoppe (1997) ao apresentar sua estrutura do VDPM, esta seqüência de transformações para refinamento da malha utilizando operações de *vsplit* define uma hierarquia de vértices única (Figura 3.17), onde os nós raízes correspondem aos vértices da malha base M^0 e as folhas correspondem à malha completa M^n. Esta hierarquia permite a criação de malhas seletivamente refinadas, que não necessariamente na seqüência original $M^0 \ldots M^n$. Uma malha seletivamente refinada *M* corresponde a uma “frente de vértices” através da hierarquia de vértices (por exemplo, M^0 e M^n na Figura 3.17) que foi obtida pela aplicação incremental das transformações de *ecol* e *vsplit* dentro de um conjunto de condições. A malha seletivamente refinada *M*, também

podendo ser chamada de malha ativa, é usualmente mais simples que a malha completa M^n.
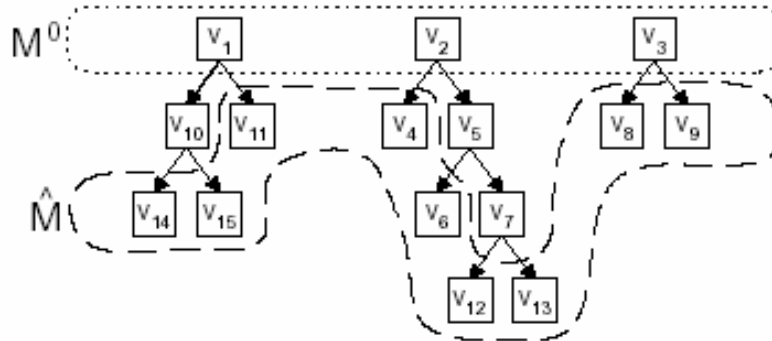
Figura 3.17 – Hierarquia de vértices definida por série de operações de *vsplit*.

Para conseguir um nível de detalhe dependente do observador (*view-dependent LOD*), a frente de vértices é varrida antes da renderização de cada quadro, sendo cada vértice podendo ser simplificado ou refinado baseados em critérios de refinamento dependentes do observador.

Por fim, Hoppe (1998) apresenta o algoritmo de VDPM provido de coerência temporal durante a criação de geomorfismos através de uma nova estrutura de dados.

Os geomorfismos permitem a eliminação dos artefatos de *popping* por meio de uma interpolação suave dentro da geometria. Quando os critérios de refinamento indicam a necessidade de uma operação de *ecol* ou *vsplit*, ao invés de realizar as transformações imediatamente, é realizado um geomorfismo pela mudança gradual da geometria de vértices durante alguns quadros.

Foi assim apresentado um método hierárquico para construir uma representação por PM de grandes terrenos. A geometria da superfície do terreno é particionada em blocos e é então utilizada uma recursão *botton-up* para simplificar e unir blocos de geometrias. Iniciando no nível mais baixo, cada bloco vai sendo simplificado pela aplicação sucessiva de uma seqüência de operações de *ecol* definidas pelo erro de aproximação mais baixo (Figura 3.18). Para evitar dependências entre blocos adjacentes, os vértices das bordas dos blocos não são alterados. Depois as malhas simplificadas resultantes são reunidas 2x2 e o processo é repetido para o próximo nível.
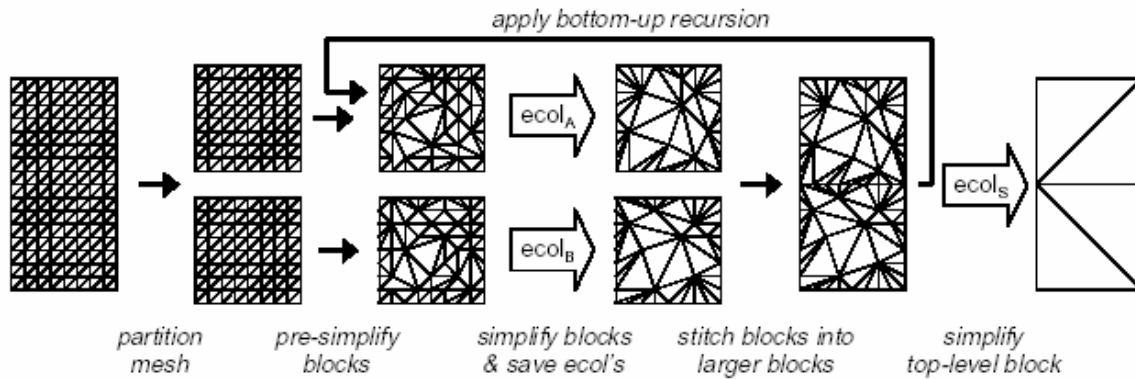
Figura 3.18 – Simplificação hierárquica baseada em blocos

Desta forma, a representação por PM define uma seqüência contínua de aproximações além de suportar transições visuais suaves (geomorfismos) entre aproximações, permite a transmissão progressiva e realiza uma efetiva compressão da malha.

Por ser baseado em TIN, o algoritmo de Hoppe além de ser complexo, demanda elevada capacidade de armazenamento e sofre de muitas limitações, perdendo em generalidade.

CAPÍTULO 4

MAPEAMENTO DE TEXTURAS

O mapeamento de texturas tem sido utilizado para dar maior realismo às imagens geradas no computador (Catmull, 1974) (Gardner, 1984). Nos últimos anos as técnicas de mapeamento de textura vêm sofrendo significantes avanços, saindo do domínio dos softwares de renderização para dentro dos hardwares gráficos.

A utilização de texturas em cenas para a visualização de simulações ou na animação computadorizada contribui para diminuir a complexidade geométrica ao mesmo tempo em que se aumenta o realismo e acrescenta detalhes. Por exemplo, ao se desenhar um muro, cada tijolo teria que ser desenhado separadamente e, mesmo assim, os tijolos apareceriam regulares demais para aparecer realísticos. Se, no entanto, for utilizado o mapeamento de textura, pode-se desenhar todo o muro como um único polígono com a imagem (podendo ser mesmo uma foto) de um muro de tijolos estampado sobre ele.

A utilização de texturas para mudar a aparência de superfícies pode ser utilizada para diversos fins: mapear cores, adicionar reflexão especular (*environment maps*), causar a perturbação do vetor normal (*bump mapping*), modular a opacidade de uma superfície translúcida (*trasparency mapping*), modelar a distribuição de luz (*illumination mapping*), entre outros exemplos (Heckbert, 1989).

A maior parte da pesquisa no mapeamento de texturas está interessada nas seguintes áreas (Cline e Egbert, 1998):

- Técnicas de anti-aliasing;
- Aceleração na renderização de texturas;
- Novas aplicações do mapeamento de texturas e
- Síntese de texturas.

4.1– CONCEITOS E DEFINIÇÕES

Antes de se falar em mapeamento de texturas é necessário definir alguns sistemas de coordenadas. O espaço da textura é o sistema de coordenadas 2D das imagens de texturas. O espaço do objeto é o sistema de coordenadas 3D no qual os elementos geométricos de um objeto ou cena estão definidos. Tipicamente, um polígono é definido, no espaço do objeto, pelo conjunto de coordenadas de cada um de seus vértices. O espaço da tela é o sistema de coordenadas do dispositivo que vai mostrar a cena como, por exemplo, o monitor do computador. O espaço de tela pode ser 2D ou 3D. Quando se

utiliza o 3D é o sistema de coordenadas da perspectiva com coordenadas dos pixels (x,y) e da profundidade z (utilizado para *z-buffering*) e o 2D é um subconjunto do 2D sem a dimensão *z* (Heckbert, 1989).

O espaço da textura é um espaço coordenado paramétrico que pode ser unidimensional (1D), 2D ou 3D. O mapeamento mais comum é o bi-dimensional, que é utilizado para definir a parametrização de uma superfície e para descrever a transformação entre o sistema de coordenadas da textura e o sistema de coordenadas da tela. A correspondência entre o espaço 2D da textura e o espaço 3D do objeto é chamada parametrização de superfície, enquanto que o mapeamento do espaço 3D do objeto para o espaço 2D da tela é a projeção definida pelas transformações do modelo e da câmera (Figura 4.1).
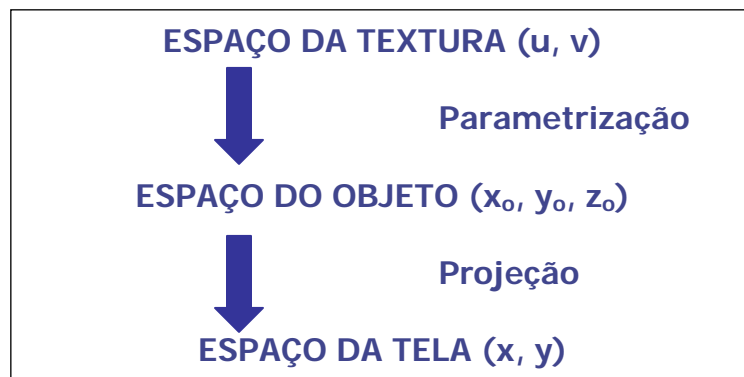
Figura 4.1 – Composição do mapeamento: parametrização e projeção.

Teschner e Henn (1995) definiram que os três componentes básicos necessários para um sistema de mapeamento de textura são:

- A textura, que é definida no espaço da textura;
- A geometria em 3D, definida por vértices;
- Uma função de mapeamento que associa a textura à descrição por vértices do objeto 3D.

Analogamente ao pixel (*picture element*) no espaço da tela, o menor elemento endereçável no espaço da textura é chamado de texel (*texture element*). A Figura 4.2 mostra um exemplo de como uma textura que é definida no espaço 2D da textura e é mapeada para o espaço 3D do objeto.
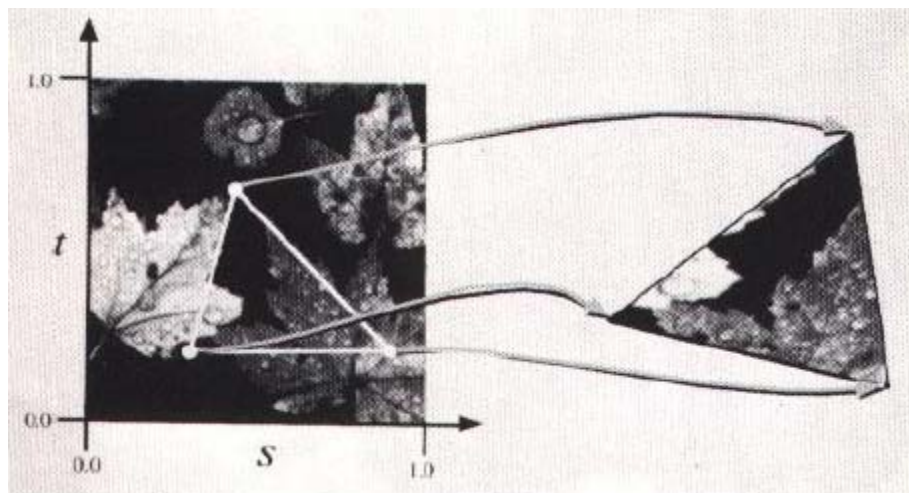
Figura 4.2 – O princípio básico do mapeamento de textura.

O mapeamento de texturas pode ser utilizado para definir outros parâmetros da superfície sendo mapeada além da cor. O trabalho de Haeberli e Segal (1993) apresenta um apanhado de aplicações que envolvem o mapeamento de textura, projeção de texturas e deformação da imagem. Teschner e Henn (1995) abordam o uso de texturas nos diversos problemas de visualização nas áreas técnica e científica como forma de representar e analisar grandes quantidades de dados experimentais ou provenientes de simulações.

4.2– MIP-MAPPING

O emprego do mapeamento de textura para dar mais realismo às imagens criadas no computador acaba trazendo outros problemas. Quando uma imagem é mapeada em um objeto, a cor de cada pixel do objeto é modificada pela cor correspondente da imagem. Primeiramente a imagem precisa ser distorcida e alterada para combinar com qualquer distorção (como a causada pela perspectiva) no objeto projetado que está sendo mostrado. Depois, a imagem alterada é filtrada para remover componentes de alta freqüência que poderiam levar a artefatos de aliasing. E, por fim, é realizada uma nova amostragem para obter a cor desejada a ser aplicada no pixel sendo texturizado.

Os problemas de artefatos de aliasing são visíveis nas texturas em polígonos que estão a alguma distância do ponto de vista do observador. Ao se movimentar rapidamente por um mundo virtual esses artefatos aparecem como brilhos ou flashes na superfície da textura. Ou, se o ponto de vista do observador é fixo, os artefatos aparecem como padrões indesejados na superfície texturizada (Figura 4.3-a) (Flavell, 1998).

A filtragem da textura para atenuar os artefatos de aliasing requer o acesso à memória de textura, que é uma operação que demanda tempo e, por isso, deve ser minimizada. Para tanto foi desenvolvido uma metodologia que utiliza texturas pré-filtradas popularmente conhecido como MIP-Mapping (Figura 4.3-b) (Williams, 1983).

(a) (b)
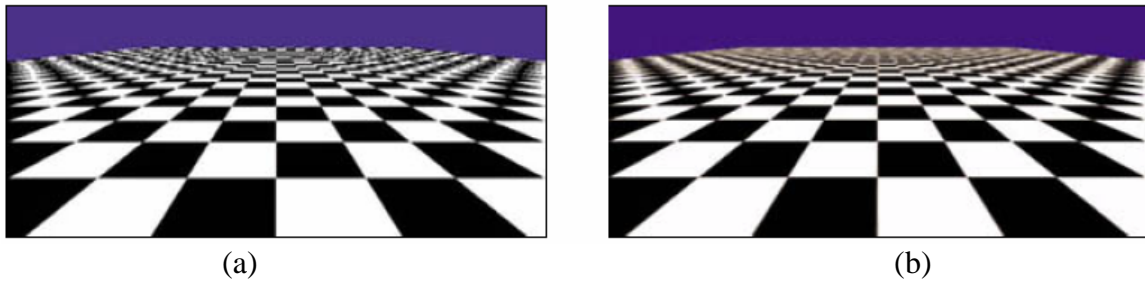
Figura 4.3 – Padrão xadrez: (a) sem MIP-Map e (b) com MIP-Map.

O termo MIP vem do latim *multum in parvo* que quer dizer muitas coisas em um pequeno espaço. A técnica do MIP-Mapping consiste de uma série de imagens pré-filtradas de resoluções decrescentes. O processo parte da imagem de maior detalhamento, conhecida como textura base ou nível 0. Os níveis seguintes são gerados a partir de seus antecessores, tirando uma média de grupos de quatro texels vizinhos para formar a o texel correspondente na imagem do nível seguinte até se chegar a um nível composto por um único texel. As dimensões iniciais da textura devem ser uma potência de 2. Desta forma, cada nível possui metade da resolução do nível anterior (Figura 4.4-a), formando uma pirâmide de resolução (Figura 4.4-b).

(a) (b)
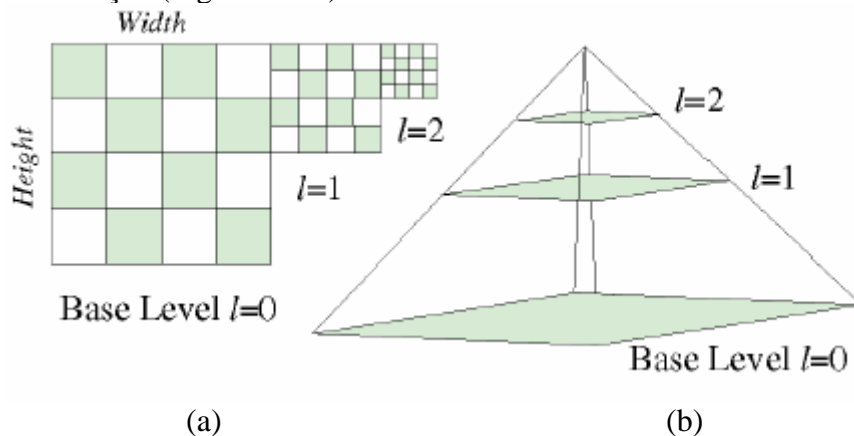
Figura 4.4 – (a) Requerimento de armazenagem; (b) Representação da estrutura piramidal de um MIP-Map.

Ao mapear a textura utilizando MIP-Maps, as imagens de maior resolução, isto é, de menor nível na pirâmide, cobrem partes menores do modelo geométrico, enquanto que os níveis de menor resolução cobrem as partes maiores correspondentes que estão mais distantes (Figura 4.5).
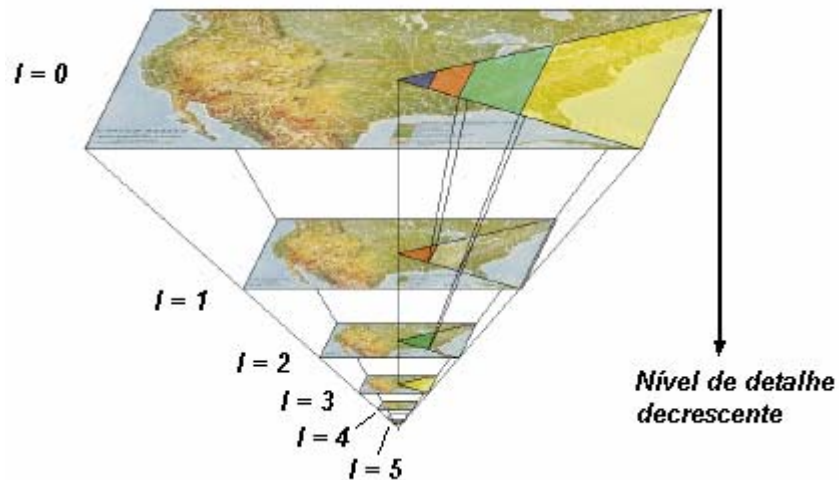
Figura 4.5 –O nível de detalhes diminui a medida que o nível do MIP-Map aumenta.

A determinação da distância onde se dará a troca entre níveis do MIP-Map é crucial: se for muito próximo do observador a imagem aparecerá borrada, se for muito longe os artefatos de aliasing continuarão sendo visíveis. Como pode-se perceber, a escolha dos texels a serem acessados em um MIP-Map está sujeita a variações nos vários fatores que controlam a projeção do pixel na tela. Como ilustração pode-se observar a Figura 4.6 que mostra a relação entre a posição do observador e o acesso aos texels de um MIP-Map.

(a) (b) (c)
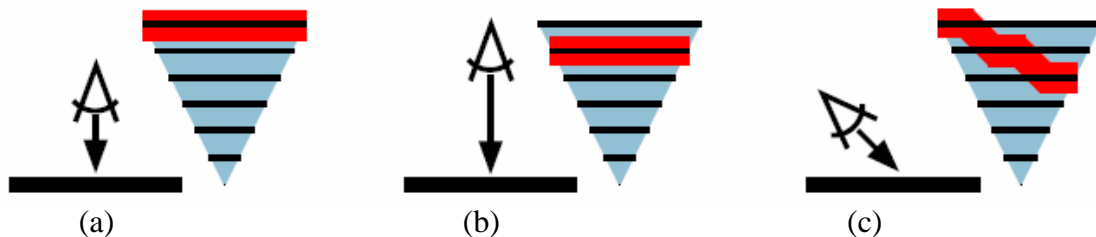
Figura 4.6 – Acesso ao MIP-Map conforme posição do observador: (a) próximo, (b) distante, (c) oblíquo.

O trabalho de Ewins *et al.* (1998) apresenta e compara vários métodos para minimização de textura e de seleção do nível do MIP-Map para o mapeamento de textura e discute a seleção de um algoritmo para implementação baseado nos fatores qualidade da imagem, custo computacional e integração com o hardware gráfico.

4.3– SISTEMA TÍPICO DE MANIPULAÇÃO DE TEXTURAS

A abordagem mais comum para o mapeamento de texturas em aplicações interativas é tratar a memória de textura como um recurso limitado e as imagens a serem utilizadas como textura como sendo recursos atômicos (Cline e Egbert, 1998).

Assim, como a memória de textura é limitada, existe um limite no tamanho de bytes de textura que uma cena pode conter. Da mesma forma, tratando as imagens de textura como recursos atômicos, cada imagem é armazenada separadamente em disco e o programa de visualização gera os MIP-Maps quando as texturas são carregadas, não sendo armazenados em disco.

Durante a execução do programa o usuário deve aguardar que cada imagem a ser utilizada como textura em uma cena seja carregada antes que possa realizar qualquer interação. A Figura 4.7 mostra a forma típica de como se dá o carregamento de texturas em aplicações de tempo-real.
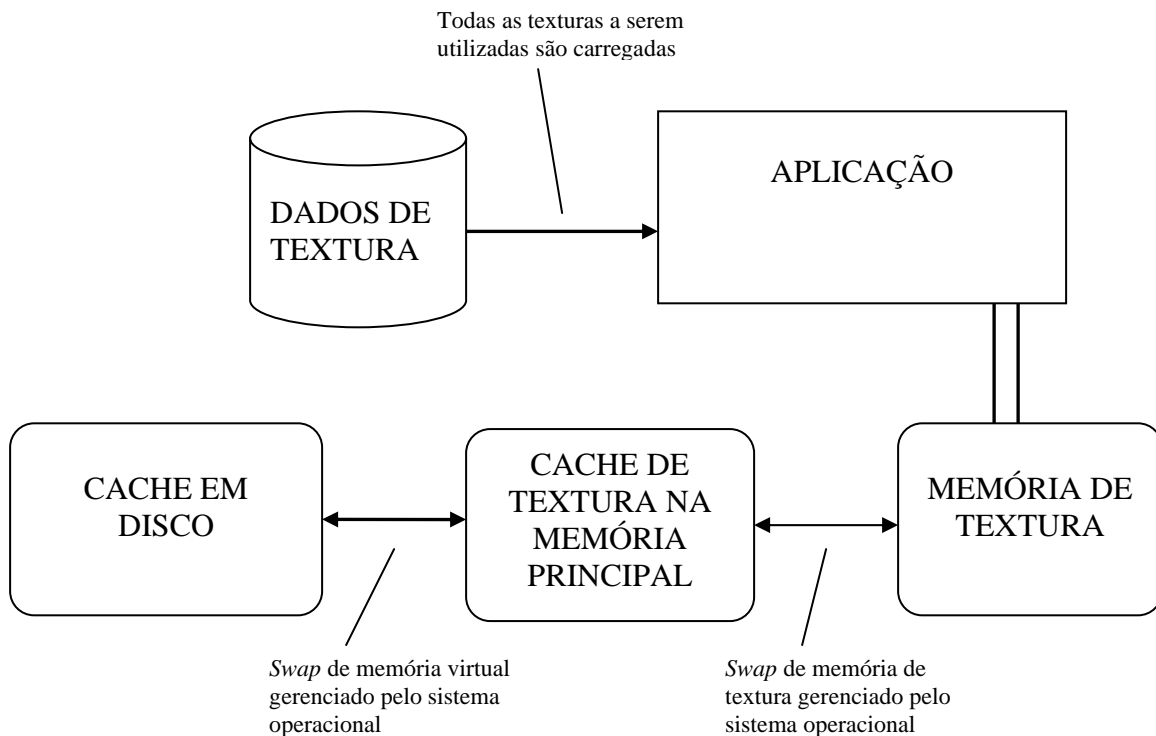
Figura 4.7 – Um sistema típico carregamento de texturas.

Quando necessidade de textura excede o tamanho da memória de textura, começam a aparecer gargalos para aplicações interativas em tempo real:

- Acesso a disco ou rede: o usuário deve aguardar enquanto é realizada leitura e transferência dos dados pela aplicação;
- Tradução da textura: se a textura é armazenada comprimida, ela deve ser descomprimida; se alguma de suas dimensões não é um par, ela vai ter que ser redimensionada; MIP-maps devem ser criados a partir da imagem original;
- Aumento de acesso à memória principal: a textura normalmente é colocada na memória de textura do hardware gráfico. Se a memória de textura não é suficiente, o sistema operacional armazena na memória principal. A interação pode ser prejudicada se houver numerosas transferências de dados entre a memória de textura e a memória principal;
- *Swapping* de memória virtual: se a textura extrapolar os limites da memória principal, o sistema operacional vai utilizar a memória virtual em disco, o que vai acarretar um tempo aguardando enquanto a textura é carregada do disco.

CAPÍTULO 5

GERENCIAMENTO DE GRANDES TEXTURAS

Controlar o nível de detalhe de uma imagem ou de uma animação onde os objetos são vistos nas mais variadas distâncias é um problema ainda bastante pesquisado nos dias de hoje. Já as técnicas de mapeamento de grandes texturas para a visualização de terrenos em tempo real continua sendo um campo fértil para o desenvolvimento de novas metodologias.

Ao final deste capítulo serão apresentadas resumidamente algumas metodologias propostas para lidar com grandes texturas em sistemas onde a memória de textura é um recurso limitado.

5.1– GERENCIAMENTO DE TEXTURAS

A medida em que houve o aumento de interesse na utilização de texturas, surgiram várias metodologias para superar as limitações impostas pelo hardware. Entre essas metodologias estão inclusas a compressão de texturas e o cache de texturas.

5.1.1 – COMPRESSÃO DE TEXTURAS

A compressão de texturas, em condições ideais, deve-se dar na memória de textura, aumentando sua capacidade e possibilitando ao hardware a diminuição da troca de dados com outros dispositivos. Uma desvantagem desses métodos é a necessidade de suporte do próprio hardware. Um exemplo pode ser encontrado no sistema Talisman (Torborg e Kajiya,1996), que emprega uma compressão similar ao JPEG em hardware e que mantém as texturas comprimidas na memória de textura.

As abordagens em software trabalham com a compressão na memória principal, reduzindo a necessidade de maior troca de dados com a memória virtual. Para que sejam eficazes, os métodos de compressão de texturas baseados em software devem ser rápidos para não atrapalhar na renderização da cena. Como exemplo pode-se citar o trabalho de Beers *et al.* (1996) que utiliza quantização vetorial (*vector quantization*) para comprimir texturas que são mantidas na memória principal.

5.1.2 – CACHE DE TEXTURAS

Mesmo com a utilização de compressão, a textura pode ser maior que a capacidade de memória. Neste caso, fazer o cache de dados pode ser utilizado para gerenciar a textura. Uma grande parte dos algoritmos de cache de textura encontrados na literatura refere-se a soluções para aplicações específicas com um problema particular de cache.

Cohen-Or et al. (1996) descrevem um sistema que trabalha com foto-texturas de terreno. Os dados são recuperados em diferentes resoluções baseados na distância do observador. Em outra aplicação para visualização de terreno, Lindstrom *et al.* (1995) usa o ângulo em que as texturas são vistas para reduzir a solicitação de texturas utilizando uma métrica de distâncias. Blow (1998) descreve algumas abordagens de cache de texturas utilizadas na produção de jogos para computador, além de discutir detalhes na implementação de um cache para o gerenciamento de grandes texturas de terrenos em um jogo.

5.2– CLIP-MAPPING

Se fosse utilizada uma textura para representar a Terra com uma definição de 1 metro, seria necessário uma textura de 40 milhões por 20 milhões de texels e cujo MIP-Map ocuparia aproximadamente 11 petabytes. Como ficou bem evidenciado, a manipulação de grandes texturas vai esbarrar nas limitações do hardware, em especial na quantidade de memória de textura disponível.

Tanner, Migdal e Jones (1998) apresentaram uma nova metodologia de manipular grandes texturas e que foi denominada de CLIP-Mapping. Foi definido que um CLIP-Map é uma representação dinâmica de textura e que armazena em cachê texturas de grandes tamanhos em uma quantidade finita de memória física para a renderização em tempo real. A metodologia proposta baseia-se no fato de que a maior parte de um MIP-Map completo não seria utilizado na renderização de uma imagem.

O CLIP-Map é uma representação atualizável de um MIP-Map parcial, no qual os níveis são limitados a um tamanho máximo especificado, denominado *ClipSize*, medido em texels. Isto vai resultar em uma estrutura com a forma de um obelisco ao invés da estrutura em forma de pirâmide do MIP-Map (Figura 5.1).
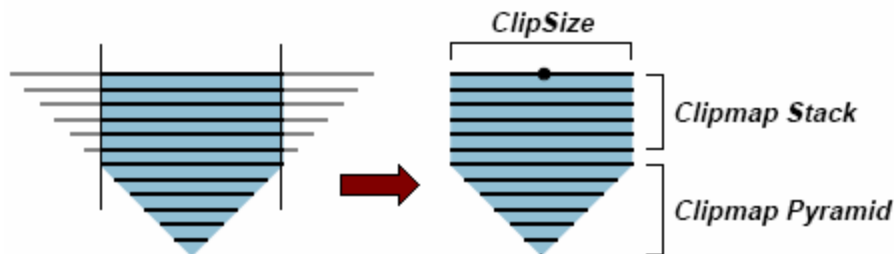
Figura 5.1 – CLIP-Map

Os níveis que possuírem um tamanho maior que *ClipSize* são cortados, porém todos os níveis mantêm o tamanho lógico e a aritmética de acesso durante a renderização do nível correspondente de um MIP-Map. O *Clipmap Stack* é o conjunto de níveis que foram cortados e possuem dimensões correspondentes a *ClipSize*, enquanto *Clipmap Pyramid* é o conjunto de níveis que possuem dimensões menores que *ClipSize* e que estão completamente contidos na memória de textura e são idênticos à porção correspondente de um MIP-Map.

O *ClipCenter* é um ponto de coordenadas arbitrárias no espaço da textura que define o centro de cada nível. Pela definição do *ClipSize* e do *ClipCenter* de cada nível, está sendo selecionada a região da textura a ser armazenada nos níveis do *Clipmap Stack* do CLIP-Map. Uma vez especificado o *ClipCenter* no nível 0, pode-se derivar o *ClipCenter* dos níveis seguintes pelo deslocamento do centro baseado na profundidade. Isso vai forçar cada nível a ficar centralizado ao longo de uma linha do centro do nível 0 até ao ápice da pirâmide. Este centro está representado pelo ponto indicado no topo da Figura 5.1.

Quando o ponto de vista do observador muda, o conteúdo da memória de cache precisa ser atualizado. O sistema tira vantagem da coerência entre quadros sucessivos para aproveitar parte do que está armazenado no cache. Considere a imagem centralizada na memória cache mais a esquerda na Figura 5.2. Um novo *ClipCenter* é especificado *d* texels acima e para a direita do antigo centro. Pode-se observar que os texels centrais permanecem inalterados (região indicada por SAME) e, desta forma, apenas as regiões superior (T), direita (R) e a quina (C) serão atualizadas, conforme a terceira situação da Figura 5.2. Utilizando endereçamento toroidal, dado novo no topo da imagem é carregado em baixo, dados na direita são carregados a esquerda e a quina superior direita é carregada com o da inferior esquerda.
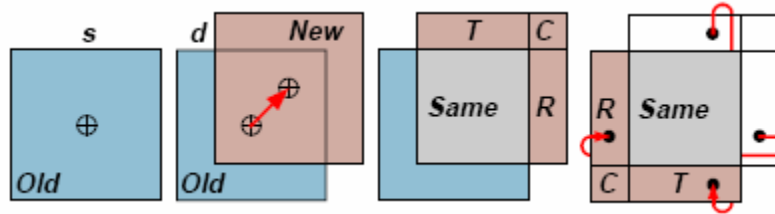
Figura 5.2 – Deslocamento e atualização da imagem.

O CLIP-Map possui alguns problemas. Um deles é o cálculo complexo envolvido para a determinação do *ClipCenter*. Outro problema é a necessidade de redimensionar a imagem para uma escala maior e na forma de um quadrado de dimensões laterais de uma potência de 2, o que leva a maior consumo de memória e a possibilidade de introduzir distorções na imagem. E, por fim, mesmo que sejam aplicadas técnicas de corte por oclusão, dada a natureza do CLIP-Map a quantidade de memória ocupada pela textura não é reduzida (Klein e Schilling, 2001).

Por fim, apesar do CLIP-Map manipular grandes texturas, sua principal desvantagem é estar disponível somente em hardwares gráficos de alta performance específicos, não sendo acessível às placas gráficas comerciais comuns.

5.3– MP-GRID

Hüttner (1998) propôs uma alternativa ao uso do CLIP-Map para a manipulação de grandes texturas. Inicialmente a imagem é redimensionada de forma que possa ser dividida em blocos que tenham suas dimensões como potências de 2. A seguir é feita o quadrilhamento (*tiling*) de toda a textura em um conjunto de MIP-Maps que são pré-calculados e armazenados em disco. Nem todos os níveis dos MIP-Maps são armazenados, uma vez que os níveis mais altos podem ser calculados rapidamente. A estrutura resultante foi batizada de *MIPmap pyramid grid* (MP-Grid).

Cada um dos MIP-Maps pode se adaptar à posição do observador apenas carregando aqueles níveis que são necessários. Para determinar quais são os níveis necessários é feita uma consulta ao volume envolvente dos polígonos que devem ser texturizados no bloco correspondente. A projeção do volume envolvente no plano da câmera do observador define o número máximo de pixels necessários para texturizar o conteúdo do volume. As dimensões são aproximadas para cima em uma potência de 2 podem ser utilizadas para determinar o nível do MIP-Map que deve estar disponível na memória (Figura 5.3).
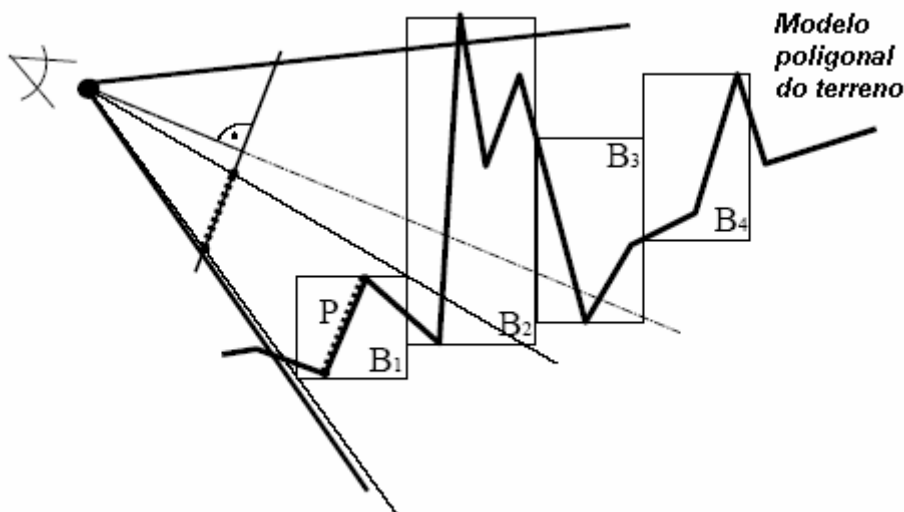
Figura 5.3 – Volumes envolventes B_i determinam os níveis do MIP-Map a serem utilizados para texturizar os polígonos de B_i.

Se houver o corte por oclusão de um conjunto de polígonos e seu volume envolvente correspondente, a adaptação do MP-Grid correspondente não é necessária e a memória de textura pode ser liberada e apenas os blocos visíveis são carregados.

O MP-Grid permite manusear texturas de tamanhos arbitrários o que constitui uma vantagem sobre o CLIP-Map. Para realizar uma comparação, considerou-se um terreno de 5000m x 5000m com o observador em (0, 0, 1000) e olhando na direção (2500, 2500, 0), a câmera possui resolução de 1024 x 1024 pixels no espaço de tela e um ângulo de visão de 45 graus. A textura utilizada possui 5120 x 5120 pixels a qual foi subdividia em um MP-Grid de 5 x 5 com uma resolução de 1024 x 1024 pixels no nível de maior resolução, ocupando 6.164.505 bytes utilizando uma escala de cinza de 8 bits. Se fosse utilizado um MIP-Map seriam necessários 34.952.550 bytes em uma escala de cinza de 8 bits. Para um CLIP-Map, a imagem teria que ser antes convertida para uma imagem de 8192 x 8192 pixels. Utilizando um *ClipSize* de 2048, seriam necessários 13.981.013 bytes.

Para garantir que a textura de um triângulo pertença apenas a um bloco do MP-Grid, as bordas do bloco correspondente são incluídos na triangulação antes da renderização, o que pode anular uma simplificação na geometria (Klein e Schilling, 2001).

5.4–TÉCNICAS DE TEXTURIZAÇÃO PARA VISUALIZAÇÃO DE TERRENOS: UM MODELO EM MULTIRESOLUÇÃO

O mapeamento de texturas pode ser considerado uma ferramenta fundamental para o projeto visual do conteúdo do terreno. Para manipular grandes texturas em multiresolução e em múltiplas camadas assim como investigar as aplicações do mapeamento de textura para fins de visualização do terreno, Döllner, Baumann e Hinrichs (2000), apresentaram uma proposta de modelo em multiresolução de texturas como uma extensão do modelo em multiresolução da geometria do terreno.

No modelo de multiresolução da textura proposto cada textura é pré-processada resultando em um MIP-Map e em uma árvore de textura (Figura 5.4), sendo que o conjunto MIP-Map e árvore de textura constituem uma camada de textura. A árvore de textura representa retalhos de texturas em diferentes níveis de detalhes sendo que, cada retalho de textura está associado a uma malha do modelo em multiresolução da geometria do terreno. As texturas podem ter tamanhos arbitrários, podendo cobrir todo o terreno ou parte dele.

Uma árvore de textura é definida por uma árvore de nós que são chamados de retalhos de texturas, analogamente às malhas da árvore de geometria do terreno. A árvore de textura está associada ao modelo hierárquico de geometria do terreno: cada retalho de textura está relacionado a uma malha de geometria. Desta forma, a árvore de textura vai possuir uma estrutura similar à árvore de geometria correspondente. O retalho de textura vai especificar o nível do MIP-Map que cobre o domínio da malha de geometria associada.

A utilização de múltiplas camadas de texturas representando diferentes dados temáticos que podem ser combinados para implementar uma ferramenta interativa e mesmo com animação para visualização de terrenos. Conceitualmente as camadas de textura podem ser divididas em:

- Temáticas: incorpora informações de cor utilizadas para informar o conteúdo do terreno;
- Luminância: especifica as modificações de brilho a serem aplicadas em outras camadas de textura;
- Topográficas: especifica as informações de sombra para um modelo de terreno;
- Visibilidade: especifica a visibilidade de outras camadas de texturas em termos de transparência.

Usualmente as texturas temáticas e topográficas são pré-calculadas e armazenadas, enquanto as texturas de luminância e de visibilidade são geradas pela aplicação em reposta à interação com o usuário.
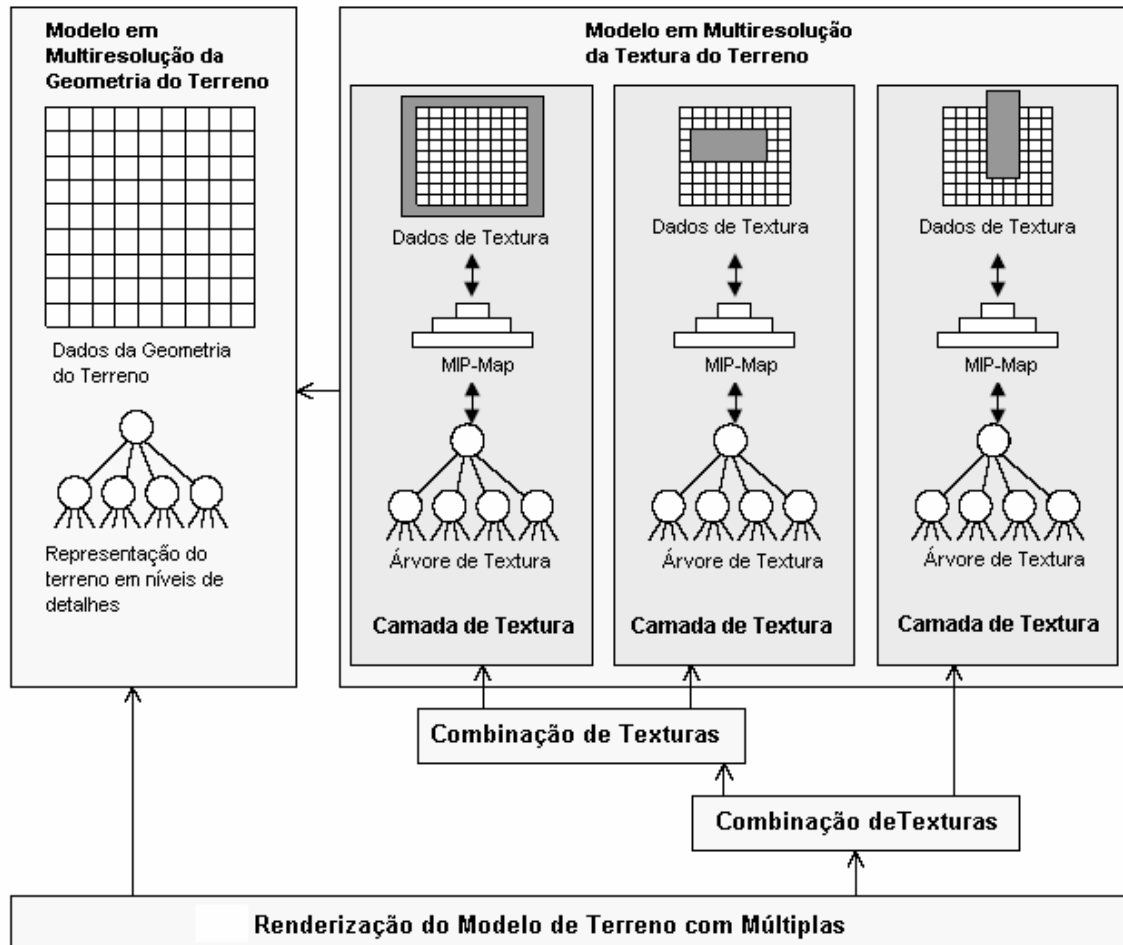
Figura 5.4 – Modelo conceitual com múltiplas camadas de textura.

O algoritmo de renderização vai levar em conta tanto os erros de aproximação geométricos como os erros de aproximação da textura. A definição dos limites de tolerância desses erros vai determinar a qualidade visual e a performance na renderização. Para tratar de mais de uma textura será necessário renderizar fazendo múltiplas passagens pelo processo, o que pode ser melhorado para aplicações de tempo real através do uso de multitexturização, já disponível nas placas gráficas comuns.

Quando mais de uma camada de textura são utilizadas, as seguintes operações para a combinação de texturas podem ser realizadas:

- *Blending*: duas camadas de textura são combinadas baseadas nos pesos fornecidos por uma textura alfa separada;
- Adição ponderada: camadas de texturas podem ser somadas sendo que os pesos de cada camada são fatores constantes;
- Modulação: duas ou mais camadas podem ser multiplicadas pixel a pixel.

Assim, múltiplos conjunto de dados temáticos podem ser superpostos no terreno através de múltiplas camadas de texturas temáticas. Um exemplo de aplicação são as lentes de textura, onde uma camada de textura de luminância (Figura 5.5-a) é utilizada para dar maior ênfase em determinada região do terreno (Figura 5.5-b). Pode-se também estabelecer a visibilidade de uma determinada região através do uso de uma camada de textura de visibilidade sobre duas camadas de texturas temáticas: a camada de visibilidade vai fornecer os pesos para combinar as camadas temáticas (Figura 5.5-c).

(a) (b) (c)
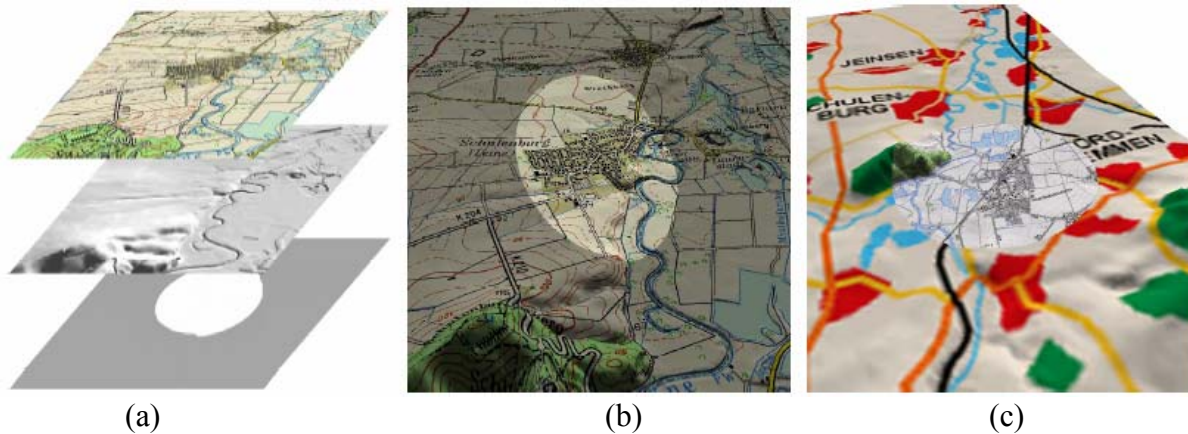
Figura 5.5 – (a) Vista conceitual das camadas de textura temática, topográfica e de luminância; (b) Lente de textura utilizada para dar ênfase em uma região; (c) Combinação parcial de duas camadas de texturas temáticas definida por uma camada de visibilidade.

Dados dinâmicos podem ser representados por uma seqüência de texturas. Na animação duas camadas de texturas estão ativas de acordo com a seqüência temporal. Essas camadas são renderizadas após a combinação ponderada com pesos α e $(1-\alpha)$ juntamente com outras camadas que por ventura existam como, por exemplo, texturas temática (Figura 5.6).
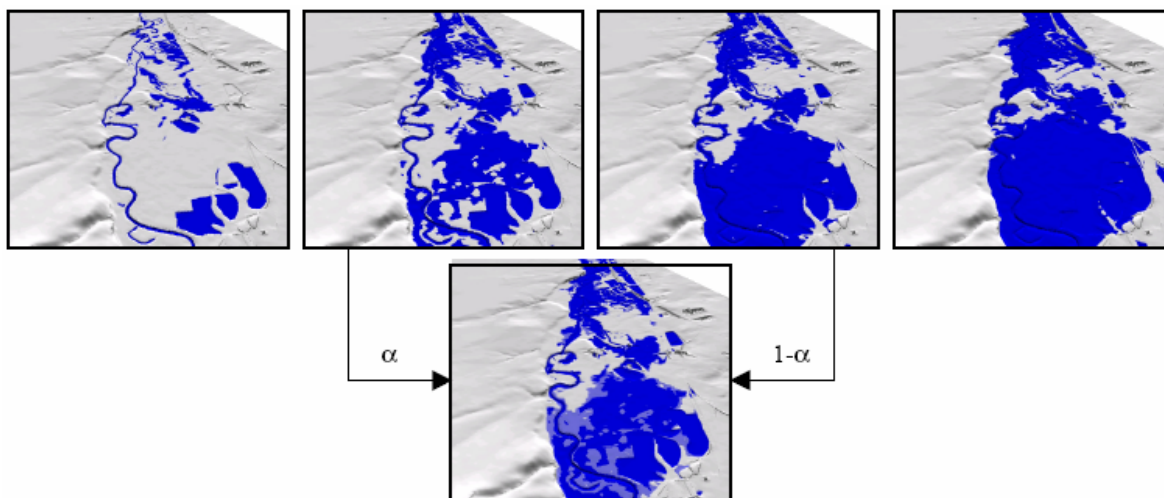
Figura 5.6 – Animação de uma inundação com interpolação pela combinação de duas camadas de textura.

A utilização de texturas topográficas leva em conta que muitas vezes o usuário percebe e reconhece a morfologia do terreno pela silhueta e pelo sombreamento do terreno. A textura topográfica é pré-calculada e depois aplicada como textura reintroduzindo detalhes topográficos que podem ter sido removidos durante o processo de simplificação da geometria (Baumann *et al.*, 2000).O cálculo da textura topográfica vai depender das propriedades da superfície, sua geometria e feições (como picos, vales, morros), fontes de luz e das regras para o sombreamento. Na Figura 5.7 a textura topográfica leva em conta o sombreamento do terreno que foi calculado pelo teste de interseção de raios da fonte de luz com o modelo do terreno. Neste exemplo a textura topográfica é modulada com a camada de textura cartográfica.

Figura 5.7 – Camada de textura topográfica com sombreamento combinada com camada de textura temática cartográfica.

CAPÍTULO 6

PROPOSTA DE TRABALHO E CONCLUSÃO

6.1 - CARTAS

Atualmente, nos sistemas militares tradicionais de Comando e Controle, o estudo do combate é baseado em mapas analógicos que descrevem cartograficamente a área do conflito e são mais conhecidos como cartas. As cartas são imagens que contêm alguma informação cartográfica e podem ser classificadas de acordo como a forma que seu conteúdo é armazenado. Assim, as cartas podem ser classificadas analógicas, ou seja, as cartas em papel, e digitais. Estas últimas podem estar no formato matricial (*raster*) ou vetorial (*vector*).

O decreto-lei número 243, de 28 de fevereiro de 1967, posteriormente alterado pelo decreto número 95.185, de 10 de novembro de 1987, estabelece as diretrizes e bases das atividades cartográficas no Brasil. Define o Sistema Cartográfico Nacional e cria a Comissão de Cartografia, incumbida de coordenar a execução da Política Cartográfica Nacional e constituída de membros de diversos ministérios tais como Defesa, Agricultura,Interior, Ciência e Tecnologia, entre outros.

No Brasil os principais órgãos responsáveis pela produção cartográfica nacional e pelo mapeamento sistemático do espaço terrestre geográfico brasileiro são o Instituto Brasileiro de Geografia e Estatística (IBGE) e a Diretoria de Serviço Geográfico (DSG), este último pertencente ao Exército Brasileiro. A cartografia náutica destinada a gerar informações para a segurança da navegação está a cargo da Marinha, enquanto a cartografia aeronáutica, destinada ao uso na navegação aérea, está a cargo da Aeronáutica.

A DSG é o órgão de apoio técnico-normativo subordinado à Secretaria de Tecnologia da Informação (STI) incumbido de superintender, no âmbito do Exército Brasileiro, as atividades cartográficas relativas à elaboração de produtos, suprimento e manutenção de material, e as decorrentes de convênios estabelecidos com órgãos da administração pública. Estão subordinadas à DSG a 1ª Divisão de Levantamento (DL), a 3ª DL, a 4ª DL, a 5ª DL e o Centro de Cartografia Automatizada do Exército (CCAuEx). A Figura 6.1 mostra as regiões de responsabilidade de cada órgão.

Cabe à DSG normalizar as escalas de 1:25.000 até 1:250.000. Estas escalas representam os níveis de detalhes necessários às cartas para o planejamento das operações. Uma carta de 1:50.000 cobre uma área correspondente a quatro cartas de 1:25.000, assim como uma carta de 1:100.000 cobre uma área correspondente a quatro cartas de 1:50.000.
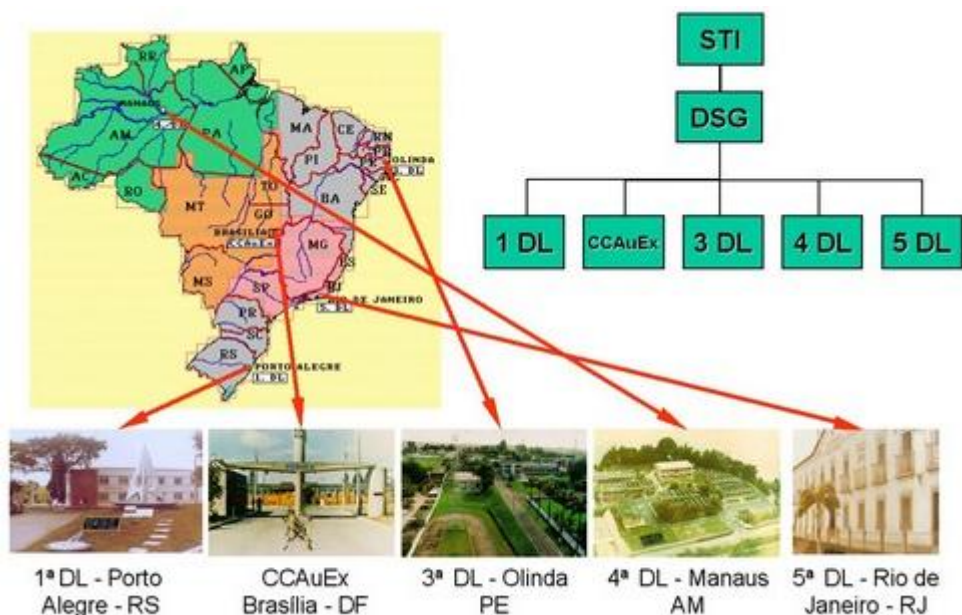
Figura 6.1 – Organização do serviço geográfico do Exército.

6.2 – LEVANTAMENTO DO PROBLEMA

A maior parte do território nacional encontra-se mapeada. No entanto, apenas em um período relativamente recente é que houve o crescimento da demanda por produtos digitais. Assim, a maior parte das cartas brasileiras ainda estão em papel.

Em algumas situações a utilização da carta em papel é de grande interesse para a montagem de mosaicos que abrangem toda a área de operação de uma força, cobrindo essa área com cartas de maior detalhamento (Figura 6.2) como, por exemplo, no planejamento de operações ou em jogos de guerra.
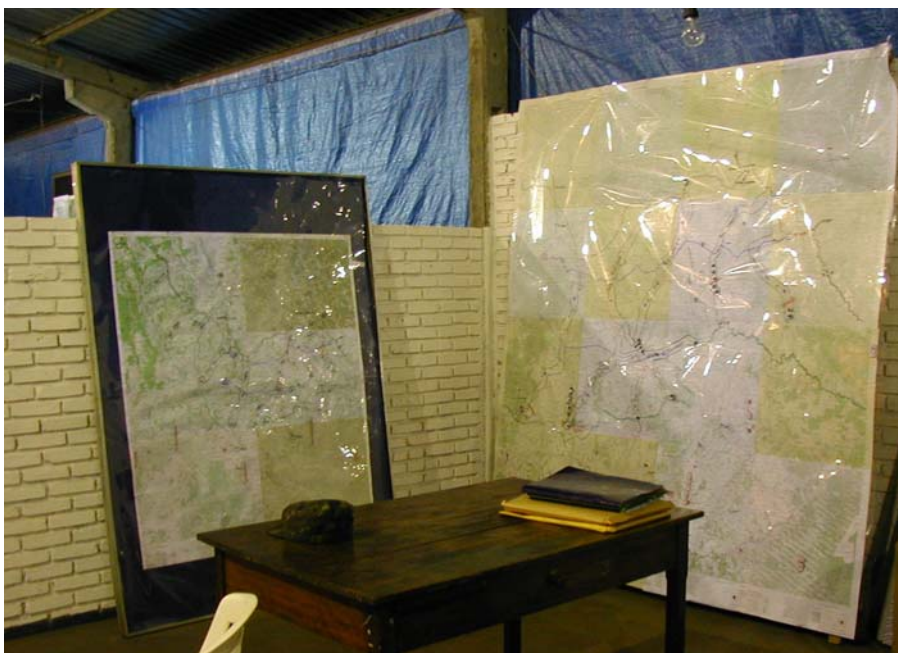
Figura 6.2 – Exemplos de mosaicos utilizados no planejamento de operações.

Além da completa mapoteca em papel cobrindo o território nacional, os planejadores já estão acostumados a manejar as cartas em papel, interpretando rapidamente suas simbologias e notações. Converter a carta em papel em uma imagem matricial é uma tarefa simples, estando as Divisões de Levantamento equipadas com o equipamento necessário. Portanto, nada mais natural do que, em um primeiro momento, desenvolver um sistema de visualização de terreno para o planejamento de operações e como ferramenta de auxílio à tomada de decisões baseado nas cartas matriciais disponíveis.

A conversão da carta de papel em uma imagem matricial traz um grande problema para a sua subseqüente utilização: o tamanho da imagem. Como exemplo, tomando-se a carta mostrada na Figura 6.3, em papel suas dimensões são de uma folha padrão A0 (1188 x 840 mm) e, após sua digitalização, tem-se uma imagem no formato TIF com as seguintes características:

Tamanho do arquivo: 58.151.864 bytes
Dimensões da imagem: 6891 x 8421 pixels
Pixels por polegada: 300
Cores: 256
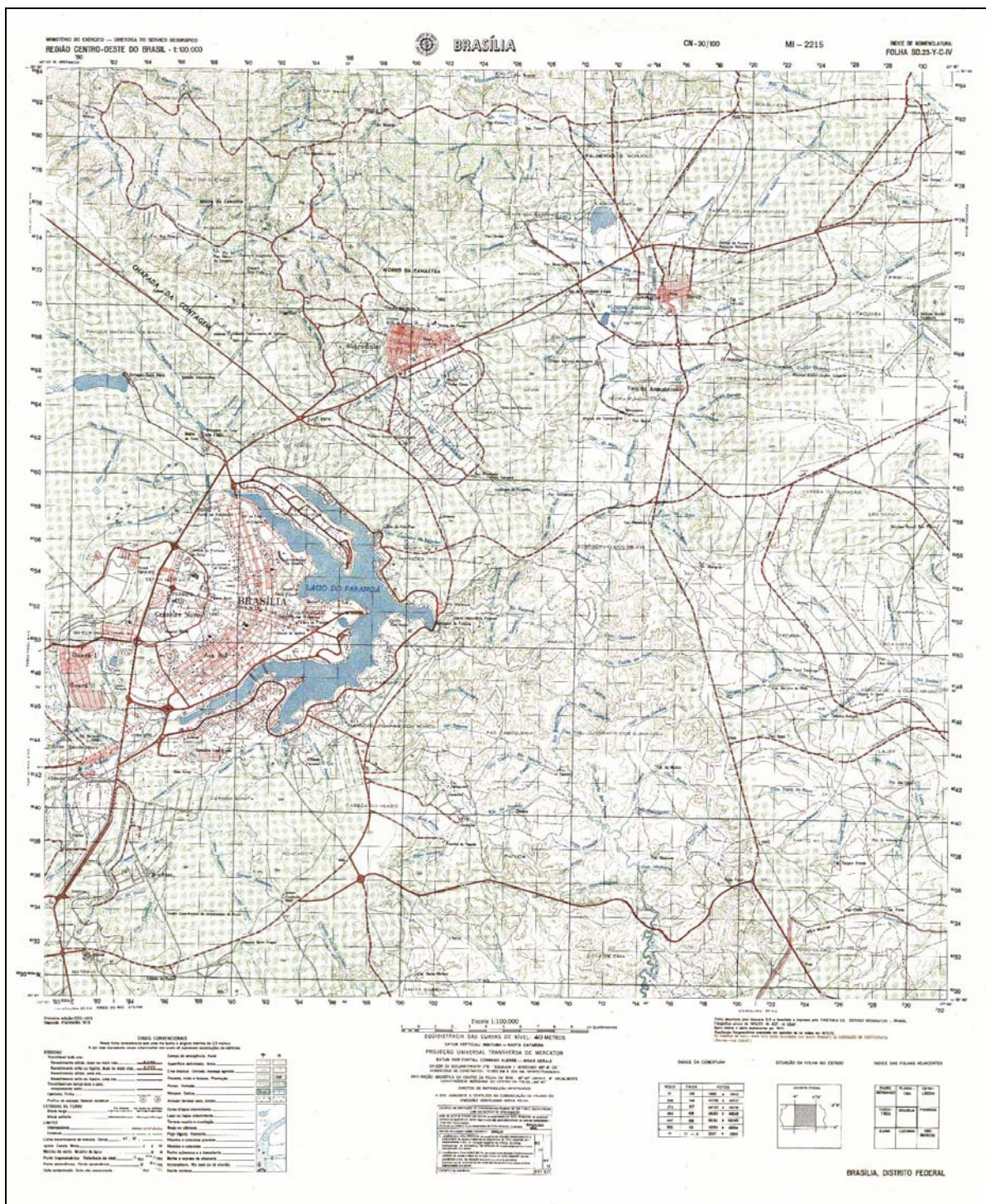
Figura 6.3 – Exemplo de carta digitalizada (escala 1:100.000).

Em outra situação pode ser necessária a visualização de mais de uma textura sobre uma mesma área. Dados cartográficos como, por exemplo, estradas, vegetação, hidrografia, tipo de solos, poderiam estar representados separadamente em diversas imagens que seriam então visualizadas separadamente de forma alternada ou como uma combinação conforme a necessidade do usuário

Adiciona-se a isso a necessidade de gerenciar outras texturas como, por exemplo, as que vão sobre outros elementos que fazem parte da representação do campo de batalha: carros de combate, bandeiras representativas das unidades, etc.

Em termos de implementação de um sistema de visualização de terrenos para utilização em Comando e Controle, deseja-se que o mesmo seja compatível com a realidade brasileira. Os equipamentos que vão manipular dados de geometrias de terrenos e de texturas devem ser computadores comerciais comuns, sem a necessidade de sistemas especializados proprietários. Por isso, a biblioteca OpenGL foi escolhida como ferramenta para o desenvolvimento da parte gráfica.

Desde que foi lançado em 1992, o OpenGL tem sido rapidamente adotado como a API (*Application Programming Interface*) gráfica para aplicações em tempo real, interativas e em 3D. A grande maioria das placas gráficas comerciais atuais já contém o suporte nativo ao OpenGL, sendo reconhecido por vários sistemas operacionais. Por ser amplamente difundida, documentada e seu funcionamento padronizado e bem definido, o OpenGL será a biblioteca gráfica a ser utilizada no desenvolvimento do trabalho.

As especificações do OpenGL são supervisionadas pela *OpenGL Architecture Review Board (ARB)* e consiste de um conjunto de companhias interessadas em disponibilizar amplamente uma API consistente. O OpenGL permite que um fabricante forneça uma funcionalidade específica para uma tecnologia recém criada através de extensões. Essas extensões são reconhecidas através da utilização de abreviações na declaração das funções dentro da extensão (por exemplo, NV para as extensões da NVIDIA). Quando mais de um fabricante concorda em implementar a mesma funcionalidade, a extensão utiliza a abreviatura EXT. Eventualmente a *Architecture Review Board* aprova e adota a extensão transformando-a em padrão, recebendo então a abreviação ARB

Até recentemente as placas gráficas suportavam apenas o mapeamento de texturas realizadas em uma única passagem. Isto quer dizer que, se fosse necessário a visualização de duas ou mais texturas deveria ser implementada em software um sistema que fizesse a renderização em duas ou mais passagens. Já as placas gráficas atuais e as novas especificações para o OpenGL suportam múltiplas texturas em apenas uma passagem pelo processamento de texturas. Multitexturização funciona como uma pilha, com uma seqüência estágios onde uma textura vai sofrer uma série de operações e seguirá para o estágio seguinte juntamente com as instruções para combinar com a textura seguinte.

Outra limitação das placas gráficas quanto às texturas é que suas dimensões devem ser potências de 2 e limitadas ao tamanho máximo suportado pela placa.

6.3 – ESTUDO INICIAL

Está se buscado um sistema de visualização do terreno onde a maior parte das informações vão estar na textura. Esta, por sua vez, poderá ser um mosaico de cartas ou várias imagens de uma mesma região representando diferentes informações: características cartográficas (vegetação, hidrografia, etc.), cartas com outras escalas, fotografias aéreas ou de satélite.

Visando o levantamento da problemática e a avaliação das tecnologias gráficas existentes, foi implementado um modesto sistema de visualização do terreno. O sistema foi desenvolvido em C++ e utiliza o OpenGL como API para a parte gráfica. A geometria do terreno utilizada foi gerada artificialmente e constitui-se de uma grade regular de 512 x 512 pontos, sendo simplificada pela utilização do algoritmo ROAM (Figura 6.4). O ROAM foi uma adaptação da implementação desenvolvida por Chris Cookson (www.btinternet.com/~cjcookson/downloads.html).
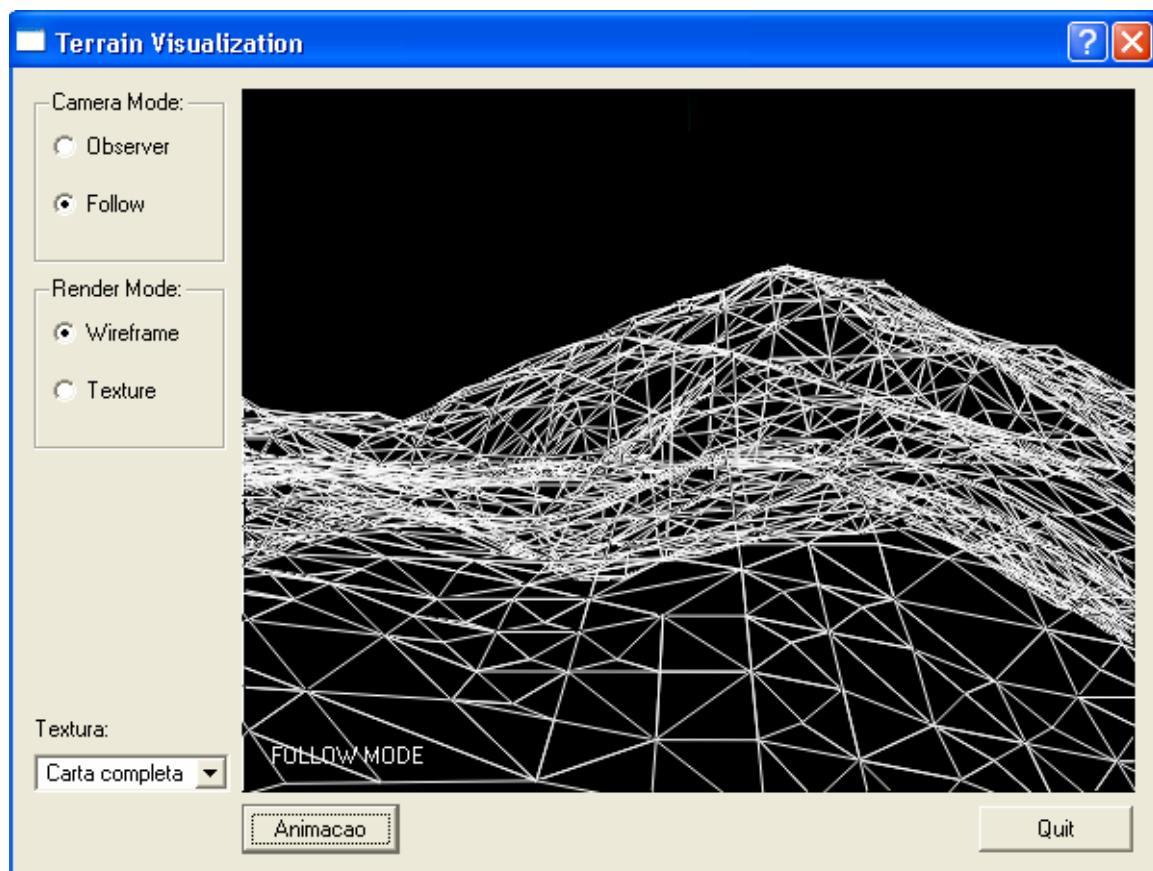
Figura 6.4 – Aplicação desenvolvida utilizando carta digitalizada como textura.

A fim de estudar as ferramentas de mapeamento de texturas fornecidas pelo OpenGL, foi feita a implementação da textura sobre o modelo geométrico fornecido pelo ROAM. Sendo assim, a textura a que foi utilizada foi gerada a partir da imagem digitalizada de uma carta (mostrada na Figura 6.3). Utilizando um software comercial de edição de imagens foi realizada uma pequena rotação de 0,3º no sentido anti-horário para corrigir a orientação da carta durante o processo de digitalização e a imagem daí resultante foram retiradas as bordas da carta. A imagem ficou então com 6318 x 6522 pixels (256 cores e 300 dpi) e com um tamanho de 13.659.718 bytes no formato TIF. Utilizar essa imagem como textura não foi possível devido às limitações de memória, sendo necessário realizar uma sub-amostragem. Como o hardware gráfico pede uma imagem que tenha dimensões potência de 2, a imagem foi redimensionada para 2048 x 2048 e para 1024 x 1024 pixels.

A sub-amostragem da imagem original vai acarretar em distorções na imagem que podem prejudicar na qualidade das informações visualizadas. Comparando um trecho da imagem original (Figura 6.5) com o das imagens sub-amostradas (Figuras 6.6 e 6.7) podemos observar que houve degradação nas imagens. A perda de informação vai acarretar no aumento da dificuldade e aumento do erro quando do processamento da carta para a filtragem de informações de características do terreno (vegetação, hidrografia, estradas, etc.). Outra conseqüência da distorção nas imagens será a perda de precisão no sistema de coordenadas geográficas.
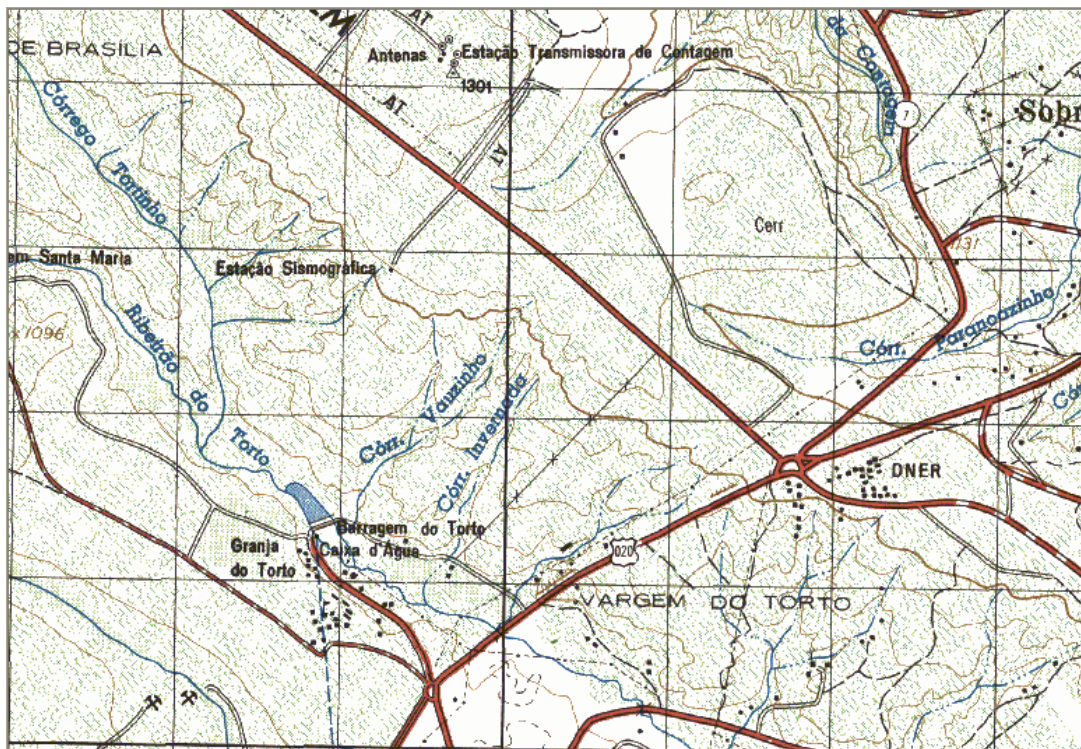
Figura 6.5 – Trecho da imagem original.
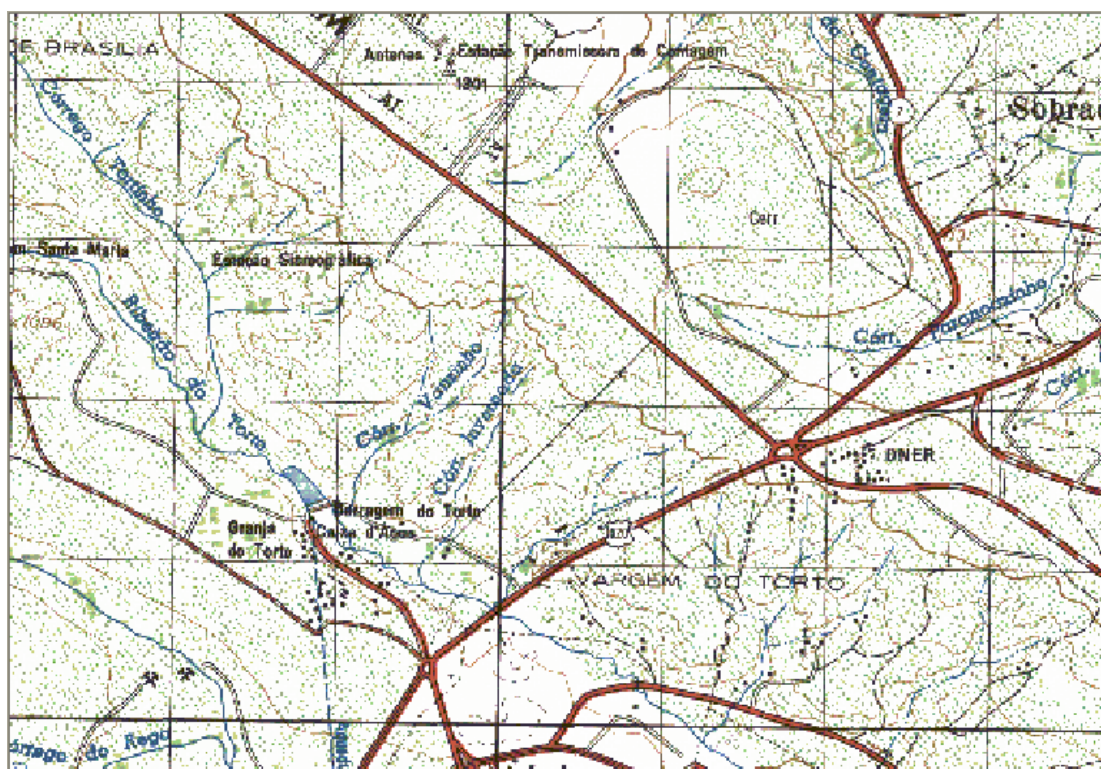
Figura 6.6 – Trecho da imagem 2048 x 2048 pixels.

Figura 6.7 – Trecho da imagem 1024 x 1024 pixels.

Nos testes, apenas a imagem de 1024 x 1024 foi carregada e mapeada com sucesso (Figura 6.8) devido às limitações da placa gráfica. Para ser utilizada como textura a imagem foi antes convertida do formato TIF para o formato JPG sendo que, nessa conversão, a imagem passou de uma palheta de 256 cores para uma de 16 milhões de cores.
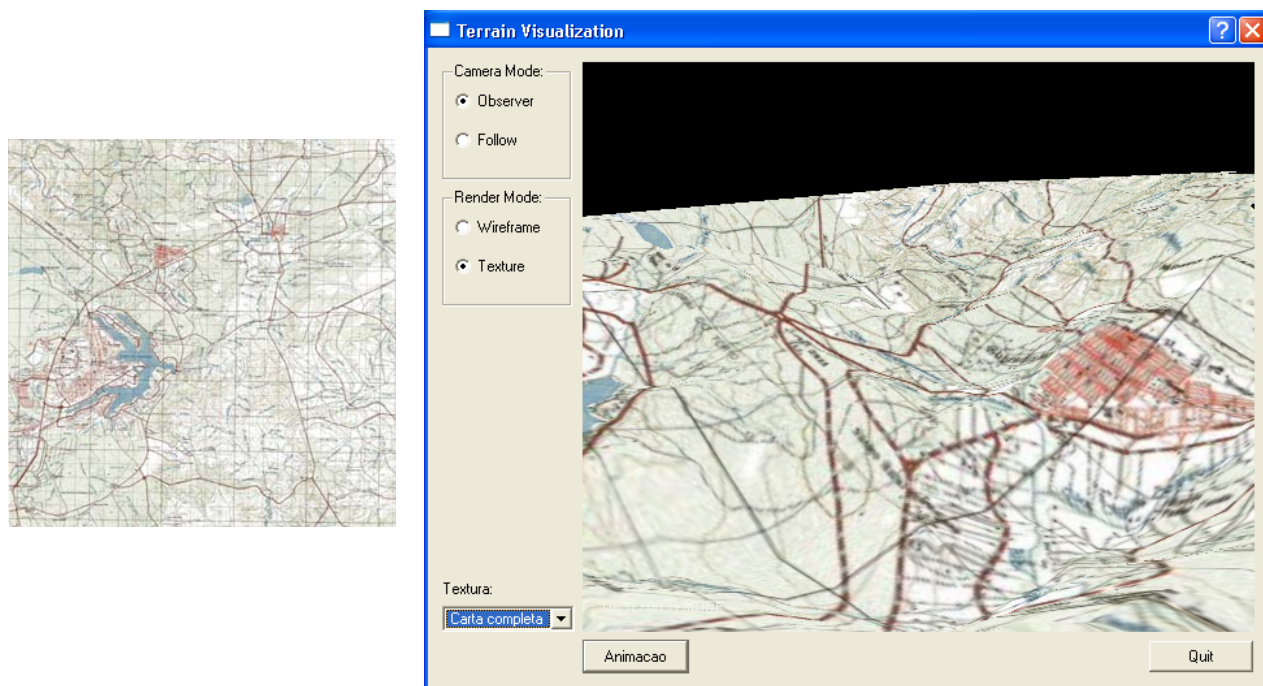
Figura 6.8 – Carta e aplicação de textura na geometria do terreno.

Considerando-se que as informações cartográficas são padronizadas, de modo geral pode-se observar que as informações sobre vegetação são representadas por tons de verde, as de hidrografia em tons de azul e as de rodovias em tons de vermelho. Realizando operações de separação em canais de cor RGB da imagem original (isto é, a carta) e analisando, filtrando e recombinando os canais, foi possível separar as informações de hidrografia (Figura 6.9), rodovias (Figura 6.10) e vegetação (Figura 6.11) de uma carta para fins de exemplificação de como essas informações poderiam ser mostradas separadamente para uma mesma região do terreno visualizado.
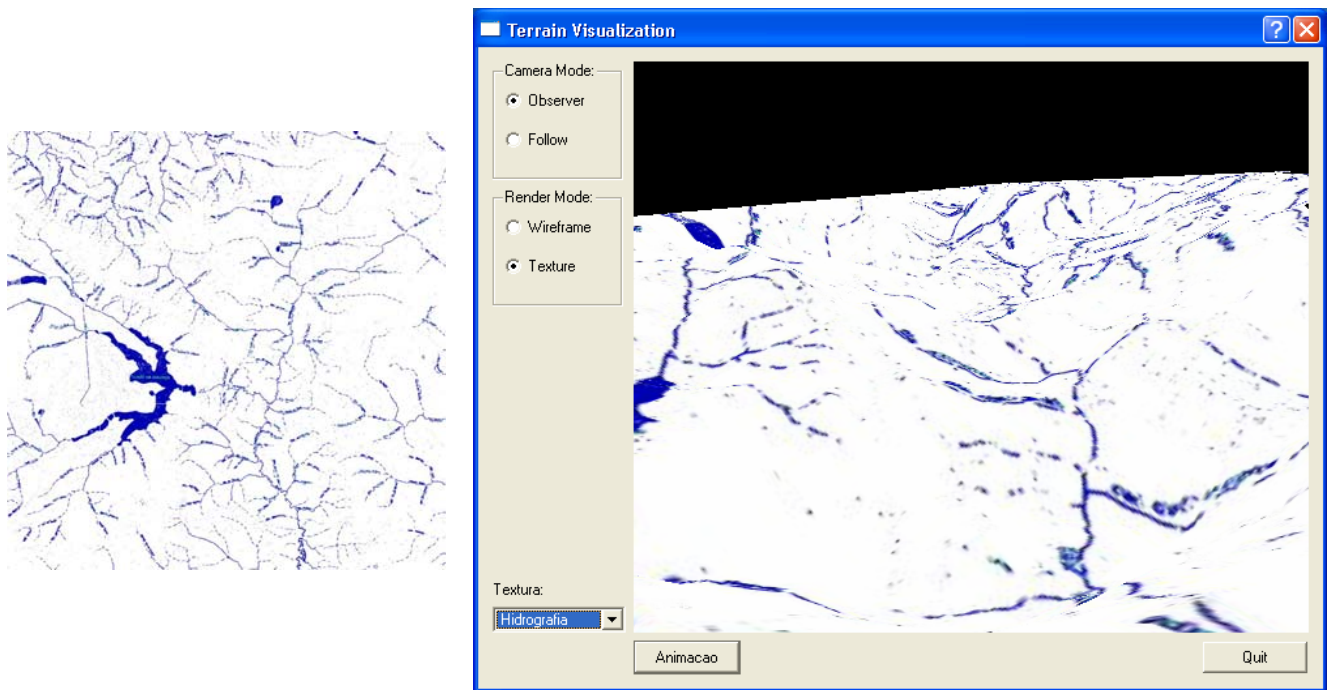
Figura 6.9 – Informação de hidrografia e textura no terreno.
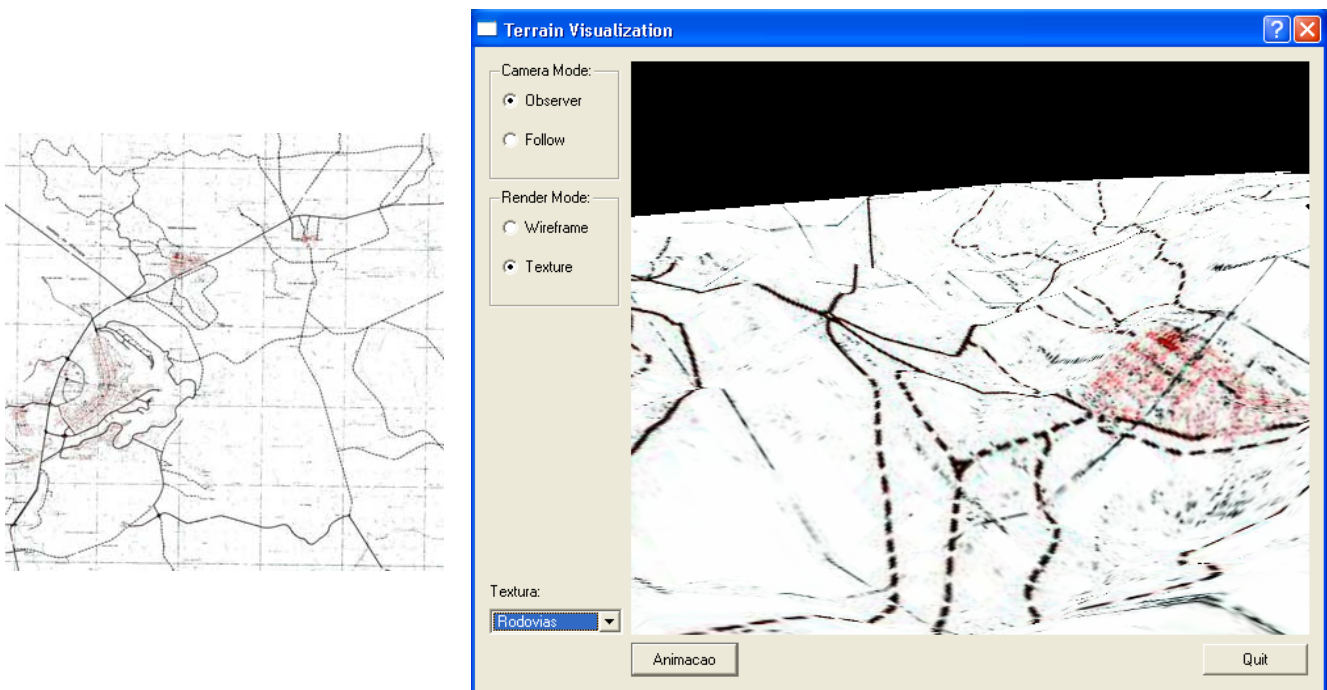
Figura 6.10 – Informação de rodovias e textura no terreno.
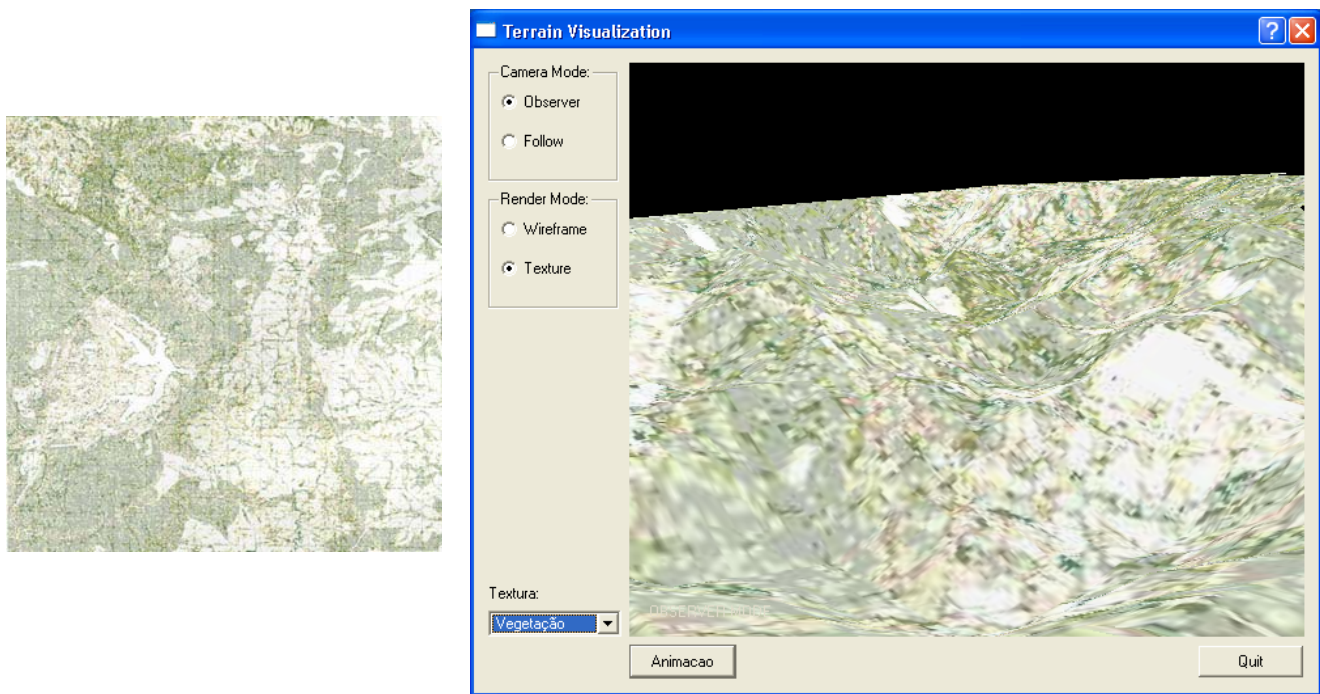
Figura 6.11 – Informação de vegetação e textura no terreno.

Buscou-se também o estudo das extensões do OpenGL, definidas nas versões mais recentes da API e implementadas em placas gráficas mais atuais, mas atendo-se somente àquelas que sejam suportadas pela maioria das placas comerciais e não específicas de um fabricante. Sendo assim, a extensão que se mostrou interessante para maiores estudos foi a ARB. Esta extensão foi formalmente revisada e oficialmente aprovada pela *OpenGL Architectural Review Board*, sendo primeiramente introduzida na versão 1.2 do OpenGL.

Dentre as extensões ARB que surgiram, a que se mostrou muito interessante foi a *GL_ARB_multitexture*. No OpenGL tradicional, o fragmento colorido entra no bloco do ambiente de textura para fazer a aplicação da textura. Já na multitextura o processo é similar, exceto que até 32 ambientes de texturas estão presentes. Para programar um ambiente de texturas em particular utiliza-se o comando:

```
glActiveTextureARB(GLenum texture);
```

onde *texture* é `GL_TEXTUREn_ARB, 0 <= n < GL_MAX_TEXTURE_UNITS_ARB`.

6.4 – PROPOSTA DE TRABALHO

Deseja-se desenvolver um sistema para o auxílio à tomada de decisões no contexto de Comando & Controle baseado em uma interface gráfica centrada na visualização do terreno em que ocorrem as operações.

De forma geral, os trabalhos se darão de acordo com o modelo mostrado na Figura 6.12. O sistema final dependerá do pré-processamento dos dados de entrada para, ao final, ser realizada a integração de todas as entradas ao sistema de visualização do terreno.
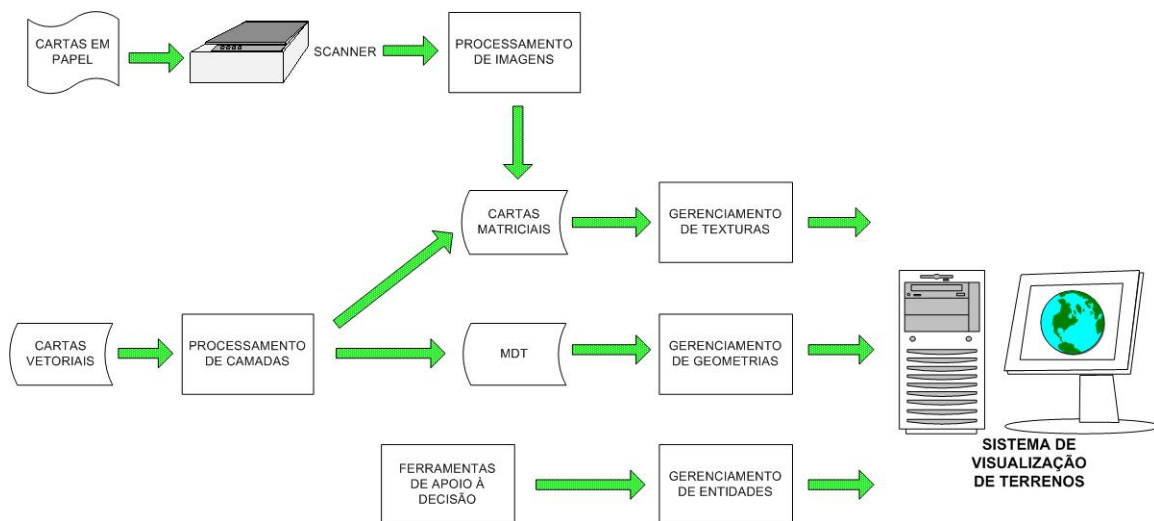
Figura 6.12 – Modelo do trabalho proposto.

As cartas vetoriais são os dados de entrada para o módulo de processamento de camadas, que será encarregado de processar as camadas de interesse que darão origem a cartas matriciais e ao MDT.

O MDT por sua vez vai ser utilizado pelo módulo de gerenciamento de geometrias para criar o modelo 2,5D do terreno. A fim de se otimizar a malha do terreno a ser visualizada, este módulo utilizará técnicas de simplificação de malhas e multiresolução de geometrias.

As cartas em papel vão passar por um processo de digitalização e então servirão de entrada para o módulo de processamento de imagens. Neste módulo, as cartas digitalizadas serão:

- Cortadas: retirar legendas e bordas, divisão em partes menores;
- Formatadas: resolução, salvos em formato de gráfico a ser definido;
- Processadas: separação de camadas de interesse tais como vegetação, hidrografia, rodovias, etc.

As cartas matriciais são a principal fonte de texturas para o modelo de geometria do terreno e serão manipuladas pelo módulo de gerenciamento de texturas. Este módulo pretende fazer face ao que foi apresentado no levantamento do problema. Será desenvolvido um mecanismo capaz de gerenciar um conjunto considerável de grandes imagens a serem utilizadas como textura em um ambiente 2,5 D, rodando em plataformas com hardwares gráficos comerciais e que seja capaz de contornar as limitações de hardware e do próprio OpenGL. Assim, propõe-se o desenvolvimento de uma metodologia que:

- Gerencie múltiplas texturas;
- Aplique multiresolução para texturas, gerenciando seus níveis de detalhes;
- Manipule e gerencie cartas em diversas escalas;
- Utilize técnicas de ladrilhamento para a partição das imagens;
- Utilize técnicas de registro de imagens para a montagem de mosaicos;
- Utilize uma ou mais estruturas de dados que possibilitem armazenar e otimizar o acesso às imagens e/ou suas partes para um determinado volume de visualização.

A técnica de ladrilhamento, dividindo o espaço em blocos e tratando-os à medida que forem necessários para a visualização dentro de um campo de visão limitado, parece ser promissora.

Outra metodologia a ser estudada será a de compressão de imagens a serem utilizadas como texturas.

As ferramentas de apoio à decisão incorporadas ao sistema terão como objetivo complementar e auxiliar na tomada de decisões. Geralmente essas ferramentas são específicas de determinadas áreas:

- Comunicações: delimitação das áreas cobertas pelas redes de rádio e determinação das melhores situações para o posicionamento dos equipamentos;
- Artilharia: delimitação das regiões de cobertura das peças de artilharia e determinação das melhores situações para o posicionamento das baterias;
- Logística: auxílio no planejamento de operações logísticas como, por exemplo, a determinação da melhor rota;
- Defesa QBR (química, biológica e radiológica): visualização de simulações de situações de contaminação QBR associado aos respectivos modelos de espalhamento.

As ferramentas de apoio à decisão poderão fornecer as entradas e o comportamento das entidades a serem controladas pelo módulo de gerenciamento de entidades o qual será responsável por fornecer outros elementos a serem visualizados dentro do ambiente de Comando e Controle como, por exemplo, posicionamento de

unidades amigas e inimigas, simbologias indicando instalações, tropas e posições importantes, etc.

Espera-se que, ao fim do trabalho, a fim de se comprovar a eficiência dos métodos e ferramentas desenvolvidos, seja criado um ambiente interativo, apropriado a uma aplicação de auxílio à tomada de decisão no âmbito de Comando e Controle de operações militares e que possa, sem a perda de generalidade, ser adaptada também para operações civis.

6.5 - CONCLUSÃO

Muitas aplicações, que vão desde Sistemas de Informações Geográficas em 3D até simuladores de vôo, utilizam-se de modelos tridimensionais de terreno. Um sistema que forneça suporte a esse tipo de aplicação de forma interativa deve possuir um mecanismo de renderização em tempo real ao mesmo tempo em que são manipuladas grandes conjuntos de dados de terreno.

As técnicas para a utilização de níveis de detalhes ainda são limitadas com respeito ao gerenciamento de grandes texturas, ao contrário do que ocorre com o uso em geometrias. A exploração interativa de dados do terreno para sua visualização com fins de análise e planejamento vai exigir o tratamento eficiente de múltiplas camadas lógicas de textura. A incorporação de múltiplas texturas nos modelos em multiresolução constitui um novo desafio.

REFERÊNCIAS BIBLIOGRÁFICAS

BAUMANN, K.; DÖLLNER, J.; HINRICHS, K. - *Integrated Multiresolution Geometry and Texture Models for Terrain Visualization* - Proceedings of the Joint Eurographics and IEEE TCVG Symposium on Visualization, pp. 157-166, May 2000.

BEERS, A. C.; AGRAWALA, M.; CHADDHA, N. - *Rendering from compressed textures* - Proceedings of SIGGRAPH'96, pp.373-378, 1996.

BLOW, J. - *Implementing a texture caching system* - Game Developer, pp. 46-56, April 1998.

BOYES, J. L.; ANDRIOLE, S. L. - *Principles of command & control* - AFCEA, 1987.

CAMPEN, A. D. - *The First Information War* - AFCEA, 1992.

CATMULL, E. - *A subdivision algorithm for computer display of curved surfaces* - Ph.D. Thesis, Department of Computer Science, University of Utah, December 1974.

CIGNONI, P.; MONTANI, C.; SCOPIGNO, R. - *A comparison of mesh simplification algorithms* - Computers & Graphics, 22 (1): pp. 37-54, 1998.

CLINE, D.; EGBERT, P. K. - *Interactive display of very large textures* - Proceedings of the IEEE Conference on Visualization'98, pp. 343-350, 1998.

COHEN-OR, D.; RICH, E.; LERNER, U.; SHENKAR, V. - *A real-time photo-realistic visual flythrough* - IEEE Transactions on Visualization and Computer Graphics, Vol 2, No. 3, pp. 255- 265, September 1996.

DE BERG, M.; DOBRINDT, K. T. G. - *On levels of detail in terrains* - Proceedings of 11th Annual ACM Symp. on Computacional Geometry, June 1995.

DÖLLNER, J.; BAUMANN, K.; HINRICHS, K. - *Texturing Techniques for Terrain Visualization* - Proceedings of the IEEE Visualization 2000, pp. 227-234, October, 2000.

DUCHAINEAU, M.; WOLINSKY, M.; SIGETI, D. E.; MILLER, M. C.; ALDRICH, C.; MINEEV-WEINSTEIN, M. B. - *ROAMing Terrain: Real-time Optimally Adapting Meshes* - Proceedings of the Conference on IEEE Visualization'97, pp. 81-88, October 1997.

DURBIN, J.; EDWARD SWAN II, J.; COLBERT, B.; CROWE, J.; KING, R.; KING, T.; SCANNELL, C.; WARTELL, Z.; WELSH, T. - *Battlefield Visualization on the Responsive Workbench* - IEEE Visualization, Proceedings of the conference on IEEE Visualization'98, pp. 463-466, 1998.

EWINS, J. P.; WALLER, M. D.; WHITE, M.; LISTER, P. F. - *MIP_Map level selection for texture mapping* - IEEE Transactions on Visualization and Computer Graphics, Vol. 4, No. 4, pp. 317-329, 1998.

FLAVELL, A. - *Run-Time MIP-Map Filtering* - Game Developer Magazine, November, 1998.
Disponível em: <http://www.gamasutra.com/features/19981211/flavell_01.htm>

FEIBUSH, E.; GAGVANI, N.; WILLIAMS, D. - *Visualization for Situational Awareness* - IEEE Computer Graphics and Applications, Vol. 20, No. 5, pp. 38-45, September/October, 2000.

GARDNER, G. Y. - *Simulation of natural scenes using textured quadtic surfaces* - Computer Graphics, Vol. 18, Number 3, pp. 11-20, July 1984.

HAEBERLI, P.; SEGAL, M. - *Texture Mapping as a Fundamental Drawing Primitive* - Proceedings of 4th Eurographics Workshop on Rendering, pp. 259-266, June 1993.

HECKBERT, P. S. - *Survey of Texture Mapping* - IEEE Computer Graphics and Applications, 6(11), pp. 56-67, November 1986.

HECKBERT, P. S. - *Fundamentals of texture mapping and image warping* - M.Sc. Thesis, Department of Electrical Engineering and Computer Science, University of California, Berkeley, USA, June 1989.

HECKBERT, P. S.; GARLAND, M. - *Survey of Polygonal Surface Simplification Algorithms* - Multiresolution Surface Modeling Course, ACM SIGGRAPH'97, 1997.

HOPPE, H. - *Progressive meshes* - Computer Graphics, Proceedings of SIGGRAPH'96, pp. 99-108, 1996.

HOPPE, H. - *View dependent refinement of progressive meshes* - Computer Graphics, Proceedings of SIGGRAPH'97, pp, 189-198, 1997.

HOPPE, H. - *Smooth View-Dependent Level-of-Detail Control and its Application to Terrain Rendering* - 1998.
Disponível em: <http://research.microsoft.com/~hoppe>

HÜTTNER, T. - *High Resolution Textures* - Proceedings of IEEE Visualization'98, October 1998.
Disponível em:

<http://davinci.informatik.uni-kl.de/vis98/archive/lbht/papers/huettnerA4.pdf>

INCE, A.N.; EVRENDILEK, C.; WILHELMSEN, D.; GEZER, F. - P*lanning and Architectural Design of Modern Command Control Communications and Information System* - Kluwer, 1997.

IMIELINSKI, T.; KORTH, H.F. - *Mobile Computing* - Kluwer, 1996.

KAUCHAK, M. - *Transforming Land Warfare – US Army Digitalization Initiative Leaps Ahead* - Armed Forces Journal, pp. 14-17, July 2001.

KLEIN, R.; SCHILLING, A. - *Efficient Multiresolution Models for Progressive Terrain Rendering* - Andreas G. Schilling (eds), Festschrift zum 60. Geburtstag von Wolfgang Straßer, pp. 109--130, Wilhelm-Schickard-Institut f. Informatik, 2001.

LEE, J. - *Comparison of existing methods for building triangular irregular network models of terrain from grid digital elevation models* - International Journal of Geographical Information Systems, 5 (3), pp. 267-285, 1991.

LINDSTROM, P.; KOLLER, D.; HODGES, L. F.; RIBARSKY, W.; FAUST, N.; TURNER, G. - *Level-of-detail management for real-time rendering of phototextured terrain* - Technical Report GTI-GVU-95-06, Georgia Institute of Technology, January 1995.
Disponível em: <http://www.cc.gatech.edu/~lindstro/papers/tr/95-14.pdf>

LINDSTROM, P.; KOLLER, D.; RIBARSKY, W.; HODGES, L. F.; FAUST, N.; TURNER, G. A. - *Real-Time, continuous level of detail rendering of height fields* - Proceedings of SIGGRAPH'96, pp. 109-118, August 1996.

MACEDONIA, M. - *Games Soldiers Play* - IEEE Spectrum, pp. 32-37, March 2002.

MELLO, F. L. - *Visualização Tridimensional do Teatro de Operações* - Tese de Mestrado, COPPE / UFRJ, Engenharia de Sistemas e Computação, Rio de Janeiro, RJ, julho 2003.

PAJAROLA, R. - *Overview of Quadtree-based Terrain Triangulation and Visualization* - UCI-ICS Technical Report no. 02-01, January, 2002.

PIERRE, G. M. - *Ambiente Integrado para Posicionamento em Operações Militares* - Dissertação de Mestrado, PUC, Departamento de Informática, Rio de Janeiro, RJ, julho 2002.

OLIVEIRA, H. J. C. - *Comando, controle, comunicação e inteligência (C^3I) nas operações de segurança* - A Defesa Nacional, 4º trim 1999.

ÖGREN, A. - *Continuous Level of Detail in Real-Time Terrain Rendering* - Master's thesis, University of UMEA, January, 2000.

RABINOVICH, B.; GOTSMAN, C. - *Visualization of large terrains in resource-limited computing environments* - IEEE Visualization, Proceedings of the Conference on Visualization'97, pp. 95-102, 1997.

RÖTTGER, S.; HEIDRICH, W.; SLUSALLEK, P.;SEIDEL, H.P. - *Real-Time Generation of Continuous Levels of Detail for Height Fields* - Proceedings of WSCG'98, pp. 315-322, 1998.

TANNER, C. C.; MIGDAL, C. J.; JONES, M. T. - *The Clipmap: A Vitual Mipmap* - Proceedings of SIGGRAPH'98, pp. 151-159, 1998.

TESCHNER, M.; HENN, C. - *Texture mapping in technical, scientific and engineering visualization* - Technical paper, SGI, August 1995.
Disponível em: <http://www.sgi.com/chembio/resources/texture/>

TORBORG, J.; KAJIYA, J. T. - *Talisman: commodity realtime 3D graphics for the PC* - Proceedings of SIGGRAPH'96, pp. 353-363, 1996.

TURNER, B. - *Real-time dynamic level of detail terrain rendering with ROAM* - Gamasutra.com, April 2000.
Disponível em: <http://www.gamasutra.com/features/20000403/turner_01.htm>

WILLIAMS, L. - *Pyramidal parametrics* - Computer Graphics, Proceedings of SIGGRAPH'83, 17(3), pp. 1-11, July 1983.

YOUBING, Z.; JI, Z.; JIAOYING, S.; ZHIGENG, P. - *A fast algorithm for large scale terrain walkthrough* - International Conference on CAD&Graphics, China, 2001.